TRANSACTIONS OF THE
AMERICAN MATHEMATICAL SOCIETY
Volume 355, Number 5, Pages 2097–2113
S 0002-9947(03)03040-X
Article electronically published on January 14, 2003

A BLACK-BOX GROUP ALGORITHM FOR RECOGNIZING FINITE SYMMETRIC AND ALTERNATING GROUPS, I

ROBERT BEALS, CHARLES R. LEEDHAM-GREEN, ALICE C. NIEMEYER, CHERYL E. PRAEGER, AND ÁKOS SERESS

Abstract. We present a Las Vegas algorithm which, for a given black-box group known to be isomorphic to a symmetric or alternating group, produces an explicit isomorphism with the standard permutation representation of the group. This algorithm has applications in computations with matrix groups and permutation groups.

In this paper, we handle the case when the degree n of the standard permutation representation is part of the input. In a sequel, we shall treat the case when the value of n is not known in advance.

As an important ingredient in the theoretical basis for the algorithm, we prove the following result about the orders of elements of S_n: the conditional probability that a random element $\sigma \in S_n$ is an n-cycle, given that $\sigma^n = 1$, is at least $1/10$.

1. Introduction

Recent advances in computing with permutation groups and matrix groups reduce a host of algorithmic problems to the constructive recognition of almost simple groups. Here we formulate constructive recognition only in the case of symmetric and alternating groups. Since we want to work with arbitrary permutation or matrix group representations of these groups, we consider the input group as a *black-box group* (cf. Section 2). Straight-line programs, which are part of the definition, are also discussed in Section 2. Recall that a randomized algorithm is *Las Vegas* if it never returns an incorrect answer; it may report failure with a small probability, prescribed by the user. In contrast, a *Monte Carlo* algorithm may return an incorrect answer, with a small probability prescribed by the user.

Definition 1.1. Let $G = \langle X \rangle$ be a black-box group, isomorphic to a finite symmetric or alternating group. We say that G is *constructively recognizable* if there are Las Vegas algorithms for the following tasks:

(i) find the isomorphism type of G (i.e., find the degree n of the natural permutation representation and decide whether $G \cong A_n$ or $G \cong S_n$);

Received by the editors September 1, 2000 and, in revised form, February 18, 2002.
2000 *Mathematics Subject Classification.* Primary 20P05, 20C40; Secondary 20B30, 68Q25.
Key words and phrases. Black-box groups, constructive recognition, symmetric groups, probabilistic method.
The third and fourth authors are partially supported by an Australian Research Council Large Grant.
The fifth author is partially supported by the National Science Foundation.

(ii) find a new set X^* of size 2 generating G and a homomorphism $\lambda\colon G \to \mathrm{Sym}(\Omega)$, specified by the image of X^*, for $\Omega = \{1, 2, \ldots, n\}$;

and, in addition, if there are deterministic algorithms for the following:

(iii) given $g \in G$, find $\lambda(g)$ and a straight-line program of length $O(n \log n)$ from X^* to g;

(iv) given $\sigma \in \mathrm{Sym}(\Omega)$, decide whether or not $\sigma \in \lambda(G)$; and, if it does, find $\lambda^{-1}(\sigma)$ and a straight-line program of length $O(n \log n)$ from $\lambda(X^*)$ to σ.

Our main result is that alternating and symmetric groups can be recognized constructively by polynomial-time algorithms. In this paper, we handle the case when the degree n of the natural permutation representation is given as part of the input, and part (i) of Definition 1.1 reduces to determining whether G is alternating or symmetric of this given degree. In the sequel [4], we shall give a modification which determines n when it is not part of the input.

An important special case is when G is the smallest dimensional faithful representation of A_n or S_n as a matrix group over a finite field. In this case, the constructive recognition of G can be done by an algorithm which is asymptotically much faster than the general procedure described in this paper. This result will appear in [5].

In order to state the main result of the present paper, we need the following notation. Suppose that the elements of G are encoded as bit-strings of length N. Let ξ be an upper bound on the time required per element to construct independent, (nearly) uniformly distributed random elements of G and let μ be an upper bound on the time required for each group operation in G.

Theorem 1.2. (*a*) *Given an integer n with $n \geq 7$, a black-box group $G = \langle X \rangle$ isomorphic to A_n or S_n, and an upper bound $\varepsilon > 0$ for the probability of failure of the algorithm, G can be constructively recognized with probability at least $1 - \varepsilon$.*

The time requirement is $O(\log \varepsilon^{-1}(\xi n + \mu |X| n \log n))$ for parts (i) *and* (ii) *of Definition 1.1. The time requirement of* (iii) *is $O(\mu n \log n)$, while constructing the straight-line program in* (iv) *is in time $O(n \log n)$ and the computation of the inverse image is in time $O(\mu n \log n)$. The data structures underlying* (iii) *and* (iv) *require the storage of $O(\log n)$ elements of G.*

(*b*) *Given an arbitrary black-box group $G = \langle X \rangle$, the algorithm described in part* (*a*) *can be used as a Monte Carlo algorithm to decide whether G is alternating or symmetric of a given degree n. The time requirement is $O(\log \varepsilon^{-1}(\xi n + \mu |X| n \log n))$.*

In the case when n is not known, our algorithm in [4] is roughly of the same asymptotic efficiency as the one indicated in Theorem 1.2.

One of the main ingredients of the algorithm in Theorem 1.2 is a result concerning the orders of elements in S_n (cf. Theorems 3.7 and 3.9). This is in the spirit of the statistical investigations of Erdős and Turán [14]. The case $m = 0$ of Theorem 1.3(a) was proved by Warlimont [18].

Theorem 1.3. (*a*) *Let $m \geq 0$ be a fixed integer. For any $\varepsilon > 0$ there exists a bound $n(\varepsilon)$ such that if $\sigma \in S_n$ for some $n > n(\varepsilon)$ then the conditional probability that σ contains an $(n - m)$-cycle, provided that $\sigma^{m!\,(n-m)} = 1$, is at least $1 - \varepsilon$.*

(*b*) *For all n, the conditional probability that $\sigma \in S_n$ is an n-cycle, provided that $\sigma^n = 1$, is at least $1/10$.*

The first polynomial-time constructive recognition algorithm for symmetric and alternating groups was described by Beals and Babai [3], and recently Bratus and Pak [7] developed a faster version. Constructive recognition algorithms for almost simple black-box groups of Lie type appeared (in chronological order) in [12], [6], [16], [9], [8], and [15].

We finish this introduction with a brief comparision of our result with [7]. Using various generalizations of Goldbach's conjecture and an oracle to compute element orders, the running time of constructive recognition in [7] is about the same as in Theorem 1.2. Eliminating the order oracle in [7] multiplies the running time by a factor n, and using a weaker form of Goldbach's conjecture which is already proven for $n > 10^{7200}$ multiplies the running time in [7] by a factor $\log^2 n$.

The paper [7] describes the case $G \cong S_n$ and sketches the necessary modifications for the case $G \cong A_n$. In this paper, we present a complete argument which works for both cases. The case $G \cong A_n$ is more complicated, and it is the more important one in applications.

The papers use similar ideas for the construction of $\lambda(g)$ for a given $g \in G$, and the asymptotic time requirement is the same. However, in [7] pre-computation and storage of $2n$ elements of G are needed, while here we compute and store less than $2 \log n$ group elements.

Unlike [7], our paper constructs straight-line programs and handles part (iv) of Definition 1.1. These are important features in most applications, and they are necessary for the proof of both parts of Theorem 1.2.

2. Preliminaries

Notation. Throughout the paper let n denote a positive integer. We denote the symmetric and alternating group of degree n by S_n and A_n, respectively. Elements of these groups will be denoted by lowercase Greek letters (except the letters λ, μ, ξ which are reserved for their special meaning in Definition 1.1 and Theorem 1.2). Elements of black-box groups will be denoted by lowercase italics. If the cycle decomposition of an element $\sigma \in S_n$ consists of m_i cycles of length i, where $0 \leq m_i \leq n$ for $1 \leq i \leq n$, then we say that σ has *cycle type* $1^{m_1} \dots n^{m_n}$. If $m_i = 0$ then we omit i^{m_i} from this expression. Note that if $n \neq 6$ then $\mathrm{Aut}(A_n) \cong S_n$, so the cycle type of $\lambda(g)$ is the same at any isomorphism $\lambda : G \to S_n$ or A_n. Hence we can talk about the *cycle type of elements of* G.

We call the set of points moved by a permutation σ the *support set* of σ and denote it $\mathrm{supp}(\sigma)$; its cardinality $|\mathrm{supp}(\sigma)|$ is called the *support* of σ. The order of the permutation σ is denoted by $|\sigma|$.

Black-box groups. A *black-box group* is a group G whose elements are encoded as 0-1 strings (bit-strings) of uniform length N, and the group operations are performed by an oracle (the "black box"). Given strings representing $g, h \in G$, the black box can compute strings representing gh and g^{-1} and decide whether $g = h$. Note that $|G| \leq 2^N$, and so we have an upper bound on $|G|$. We always let μ be an upper bound on the time for each multiplication or inversion or equality test within the group G. Clearly $\mu \geq N$.

Each string represents at most one element of G, and the same element of G may be encoded by different strings. Thus, an equivalence relation is defined on the set of bit-strings: one class C consists of those bit-strings which do not represent elements of G and the elements of G are the other equivalence classes. We do *not*

assume that we can determine whether a bit-string is in C. Given a bit-string in C as input to the multiplication or inversion oracle, the oracle may report failure or it may return a bit-string.

Straight-line programs. A *straight-line program* reaching some $g \in G = \langle X \rangle$ is a sequence of expressions $(w_1, \ldots, w_m)$ such that for each i one of the following holds:

- w_i is a symbol for some element of X,
- $w_i = (w_j, -1)$ for some j with $j < i$, or
- $w_i = (w_j, w_k)$ for some j, k with $j, k < i$,

and, if the expressions are evaluated, then the value of w_m is g. Here, $(w_j, -1)$ is evaluated as the inverse of the evaluated value of w_j, and (w_j, w_k) is evaluated as the product of the evaluated values of w_j and w_k.

A straight-line program of length m may reach an element of G which can be written only as a word of length exponential in m. In fact, Babai and Szemerédi [2] showed that, for any $G = \langle X \rangle$ and $g \in G$, there exists a straight-line program from X to g using at most $(\log |G| + 1)^2$ multiplications and inversions. Keeping track of how the new generating set X^* is constructed (cf. Definition 1.1), we obtain a constructive version of this result, though with a weaker bound on the length of such a program.

Random elements. Our methods are largely statistical. In the theoretical analysis, we assume that we can construct a sequence of independent, uniformly distributed elements of G. This sequence will be used as follows. If a certain proportion of the elements of G have a certain property, we expect to find an element with this property on repeated random selection from G.

Although uniformly distributed random elements of G are *not* available in most applications, there are adequate methods for generating random elements both from the theoretic [1] and from the practical [11] points of view. We refer to [16, Section 2.2.2] for a brief discussion of the ways in which the quality of the random element generator influences constructive recognition algorithms.

Presentations for the symmetric and alternating groups. In order to make our algorithms Las Vegas, we shall check that the elements of the new generating set constructed in Theorem 1.2 satisfy a certain presentation of the alternating or symmetric group, as appropriate.

The following presentation for S_n can be found in the book of Coxeter and Moser [13].

$$\langle s, t \mid s^n = t^2 = (st)^{(n-1)} = [t, s^j]^2 = 1 \text{ for } 2 \leq j \leq n/2 \rangle. \tag{2.1}$$

Note that $\sigma = (1, 2, \ldots, n)$ and $\tau = (1, 2)$ satisfy this presentation.

The following presentations for A_n,

$$\langle s, t \mid s^{n-2} = t^3 = (st)^n = (ts^{-k}ts^k)^2 = 1 \text{ for } 1 \leq k \leq (n-3)/2 \rangle, \tag{2.2}$$

if n is odd and

$$\langle s, t \mid s^{n-2} = t^3 = (st)^{n-1} = [t, s]^2 = 1 \rangle, \tag{2.3}$$

if n is even, are due to Carmichael [10]. Note that $\sigma = (1, 2)(3, 4, \ldots, n)$ if n is even or $\sigma = (3, 4, \ldots, n)$ if n is odd and $\tau = (1, 2, 3)$ satisfy this presentation.

Organization of the paper. In Section 3, we describe some number theoretic and probabilistic estimates which are used in the proof of Theorem 1.2. Section 4 contains the construction of the new generating set X^* (cf. Definition 1.1, part (ii)). In Section 5, we construct the data structure which supports the algorithms in parts (iii) and (iv) and summarize the proof of Theorem 1.2.

3. Number theoretic and probabilistic estimates

In this section, we collected some estimates which are needed in the proof of Theorem 1.2. The first four lemmas are of a number theoretic nature. In most cases, there are better asymptotic results but, for the algorithmic applications, we need estimates which are valid for all values of n.

Lemma 3.1. *For all n, the number of divisors of n is at most $c_1 n^{1/3}$ with $c_1 = 48/2520^{1/3} \approx 3.527$, and the sum of the divisors of n is at most $n(3+\ln n)/2$.*

Proof. For each $\delta > 0$, the number of divisors of n is at most $C_\delta n^\delta$ for a constant C_δ. An algorithm for computing the value of C_δ is described in [17, pp. 395–396]; $C_{1/3} = 48/2520^{1/3} \approx 3.527$.

We use the trivial estimate $\sum_{i=1}^{\lceil\sqrt{n}\rceil-1} i < n/2$ for the sum of the divisors less than $\sqrt{n}$. The sum Σ of divisors greater than or equal to $\sqrt{n}$ satisfies

$$\Sigma \le n + \frac{n}{2} + \frac{n}{3} + \cdots + \frac{n}{\lfloor\sqrt{n}\rfloor} < n(1 + \int_1^{\sqrt{n}} \frac{1}{t} dt) = n(1 + \frac{\ln n}{2}).$$

Adding these two estimates, we obtain the second assertion of the lemma. □

Lemma 3.2. *Let $x = np$, where p is prime, $n > p^2$, and all prime divisors of n are greater than p. Let D denote the set of divisors of x which are not greater than n. Then $|D| \le c_2 n^{1/3}$ with $c_2 = 2c_1 \approx 7.054$ and*

$$\sum_{d\in D} d \le n\,\frac{p+3-2(p+1)\ln p + (p+1)\ln n}{2}.$$

Proof. Because $p \not| n$, the number of divisors of x is twice the number of divisors of n. Thus Lemma 3.1 yields $|D| \le 2c_1 n^{1/3}$.

Also, because of the restriction on the prime factorization of n, all divisors of x except np are at most n, and $\sum_{d\in D} d = (p+1)\sum_{d|n} d - pn$. Hence it is enough to estimate the sum of the divisors of n. We use a refinement of the argument in Lemma 3.1. For the sum of divisors less than $\sqrt{n}$, we still use the trivial estimate $n/2$. However, for the larger divisors, we note that since the largest proper divisor of n is at most $n/(p+1)$, the sum Σ of divisors greater than or equal to $\sqrt{n}$ satisfies

$$\Sigma \le n + \frac{n}{p+1} + \frac{n}{p+2} + \cdots + \frac{n}{\lfloor\sqrt{n}\rfloor} < n(1 + \int_p^{\sqrt{n}} \frac{1}{t} dt) = n(1 + \frac{\ln n}{2} - \ln p).$$

Combining these estimates, we obtain the assertion of the lemma. □

Lemma 3.3. *Let $x = ny$, and let D denote the set of divisors of x which are not greater than n. Let $k > 1$. Then*

$$\sum_{d\in D} d^k \le n^k(1 + \frac{y}{k-1}).$$

Proof. All divisors $d \in D$ are of the form $d = x/s$ for some $s \geq y$. Therefore,

$$\sum_{d\in D} d^k \leq \sum_{s=y}^{x} \left(\frac{x}{s}\right)^k < \left(\frac{x}{y}\right)^k + x^k \int_y^\infty \frac{1}{t^k} dt$$
$$= \left(\frac{x}{y}\right)^k + x^k \cdot \frac{1}{y^{k-1}(k-1)} = \left(\frac{x}{y}\right)^k \left(1 + \frac{y}{k-1}\right).$$

□

Lemma 3.4. *Let $x = m!\,(n-m)$. If $n > m + m!\,m$ then the largest divisor of x not exceeding n is $n-m$.*

Proof. Suppose that there is a divisor k of x with $n-m < k \leq n$. Then $(k, n-m) \leq m$, and so the least common multiple of k and $n-m$ is at least $k(n-m)/m$. This implies $x \geq k(n-m)/m > k(m!\,m)/m > m!\,(n-m)$, a contradiction. □

The proofs of the next two lemmas are trivial counting arguments, and they are omitted.

Lemma 3.5. *For $n \geq 10$, each column of the following table lists a cycle type in the first row and the proportion of elements of that cycle type in S_n in the second row.*

cycle type	n^1	$1^1(n-1)^1$	$3^1(n-3)^1$
proportion	n^{-1}	$(n-1)^{-1}$	$\frac{1}{3}(n-3)^{-1}$

cycle type	$1^1 3^1 (n-4)^1$	$1^2 3^1 (n-5)^1$	$1^3 3^1 (n-6)^1$
proportion	$\frac{1}{3}(n-4)^{-1}$	$\frac{1}{6}(n-5)^{-1}$	$\frac{1}{18}(n-6)^{-1}$

In A_n, the proportions are 0 or twice the proportions in S_n, depending on whether the permutations with the desired cycle type are odd or even.

Lemma 3.6. *Let q be an integer in the range $n/2 < q \leq n$. The proportion of elements of S_n containing a cycle of length q is $1/q$. In A_n, this proportion is $1/q$ if $q \leq n-2$, and it is 0 or $2/q$ if $q \in \{n-1, n\}$.*

Let x be the product of the cycle lengths in a cycle type occurring in Lemma 3.5. The rest of this section is devoted to estimating the conditional probability that a randomly chosen element $\sigma \in A_n$ or S_n satisfying $\sigma^x = 1$ has the cycle type which defined x. This is the key step in the analysis of the algorithm proving Theorem 1.2.

For integers n, x, we define $T_n(x) := \{\sigma \in S_n : \sigma^x = 1\}$ and $N_n(x) := |T_n(x)|$.

Theorem 3.7. *Let m be a fixed non-negative integer. For any $\varepsilon > 0$ there exists a bound $n(\varepsilon)$ such that if σ is a uniformly distributed random permutation from S_n for some $n > n(\varepsilon)$ then the probability that $\sigma^{m!\,(n-m)} = 1$ is less than $(1+\varepsilon)/n$.*

Proof. If σ is a uniformly distributed random permutation from S_n then $\mathrm{Prob}(\sigma^{m!\,(n-m)} = 1) = N_n(m!\,(n-m))/n!$. Therefore, we have to prove the upper bound $(1+\varepsilon)n!/n$ for $N_n(m!\,(n-m))$. We can suppose that $n > 8 + 8m!\,m$. We denote the set of divisors of $m!\,(n-m)$ by D.

The basic strategy of the proof is as follows. For $\sigma \in S_n$, $\sigma^{m!\,(n-m)} = 1$ if and only if the length of each cycle of σ is a divisor of $m!\,(n-m)$. This gives us too many conditions to handle so, instead, we fix a number k and consider the set T_k^*

of those σ which satisfy the property that the lengths of the *cycles intersecting the first k points of the permutation domain* are divisors of $m!\,(n-m)$. Clearly, $N_n(m!\,(n-m)) \le |T_k^*|$. We shall compute an upper estimate for $|T_k^*|$ in terms of n, m and k (cf. (3.3) below), and we are lucky enough that k can be chosen to be a constant (depending on m and ε, but independent of n) so that for large enough n, this upper bound is less than $(1+\varepsilon)n!/n$.

Let $k \ge 3$ be fixed, let $\Pi = \{P_1, \dots, P_\ell\}$ be a fixed partition of $\{1, \dots, k\}$ into ℓ non-empty parts for some $\ell \le k$, and let $d_1, d_2, \dots, d_\ell$ be a sequence of elements from D such that $\sum_{i=1}^{\ell} d_i \le n$. First, we estimate from above the number $N(k, \Pi, d_1, \dots, d_\ell)$ of permutations $\sigma \in T_n(m!\,(n-m))$ for which σ has cycles $C_1, \dots, C_\ell$ of lengths $d_1, \dots, d_\ell$, respectively, such that $C_i \cap \{1, 2, \dots, k\} = P_i$ for $1 \le i \le \ell$.

We can choose the support set of C_1 in $\binom{n-k}{d_1-|P_1|}$ ways and then the cycle C_1 itself in $(d_1-1)!$ ways. Recursively, if $C_1, \dots, C_{i-1}$ are already defined, the support set of C_i can be chosen in

$$\binom{n-k-\sum_{j=1}^{i-1}(d_j-|P_j|)}{d_i-|P_i|}$$

ways and then the cycle C_i in $(d_i-1)!$ ways. Finally, after $C_1, \dots, C_\ell$ are defined, the rest of the permutation can be chosen in at most $(n-\sum_{i=1}^{\ell} d_i)!$ ways. Multiplying these numbers, we obtain

$$N(k, \Pi, d_1, \dots, d_\ell) \le \frac{(n-k)!\prod_{i=1}^{\ell}(d_i-1)!}{\prod_{i=1}^{\ell}(d_i-|P_i|)!} \le (n-k)!\prod_{i=1}^{\ell} d_i^{|P_i|-1}. \tag{3.1}$$

For a fixed partition Π, let $N(k, \Pi)$ denote the sum of all $N(k, \Pi, d_1, \dots, d_\ell)$ obtained above for all choices $d_1, \dots, d_\ell$ of divisors of $m!\,(n-m)$. If $\ell = 1$ then we obtain $N(k, \Pi) \le (n-k)!\sum_{d \in D} d^{k-1}$ which, by Lemmas 3.4 and 3.3, is at most

$$(n-k)!(n-m)^{k-1}(1+\frac{m!}{k-2}). \tag{3.2}$$

For $\ell \ge 2$, we use the estimates that by Lemma 3.1, a sequence $d_1, \dots, d_\ell$ from D can be chosen in at most $(8(m!\,(n-m))^{1/3})^\ell$ ways, and for each sequence trivially $\prod_{i=1}^{\ell} d_i^{|P_i|-1} \le n^{\sum_i(|P_i|-1)} = n^{k-\ell}$. Hence, since $n \ge 8 + 8m!\,m$,

$$\begin{aligned} N(k, \Pi) &\le (n-k)!\,(8(m!\,(n-m))^{1/3})^\ell n^{k-\ell} \\ &\le (n-k)!\,(8(m!\,(n-m))^{1/3})^2 n^{k-2} < 64m!\,(n-k)!\,n^{k-4/3}. \end{aligned}$$

We estimate $(n-k)!$ by

$$(n-k)! = \frac{n!}{n^k}\frac{n}{n-1}\frac{n}{n-2}\cdots\frac{n}{n-k+1} < \frac{n!}{n^k}(1+\frac{k}{n-k})^k < \frac{n!}{n^k}e^{\frac{k}{n-k}},$$

and the number of partitions of $\{1, 2, \dots, k\}$ by k^k (note that each partition can be obtained as the sets where some function $f : \{1, 2, \dots, k\} \to \{1, 2, \dots, k\}$ takes constant values). Combining these estimates with the observation that each element of $T_n(m!\,(n-m))$ is counted exactly once in $\sum_\Pi N(k, \Pi)$, we obtain that

$$N_n(m!\,(n-m)) \le n!\big((1+\frac{m!}{k-2})e^{\frac{k}{n-k}}\frac{1}{n} + 64m!k^k e^{\frac{k}{n-k}}\frac{1}{n^{4/3}}\big). \tag{3.3}$$

Given $\varepsilon > 0$, we choose k such that

$$(1 + \frac{m!}{k-2})e^{\frac{k}{k^2-k}} < 1 + \frac{\varepsilon}{2}.$$

After that, we choose $n_0 > k^2$ such that

$$64m!\, k^k e^{\frac{k}{n_0-k}} < \frac{\varepsilon}{2} n_0^{1/3}.$$

Then, for $n > \max\{8 + 8m!\, m, n_0\}$, we have $N_n(m!\,(n-m)) < (1+\varepsilon)n!/n$. □

Corollary 3.8. *Let $\varepsilon > 0$ and $n > n(\varepsilon)$.*

(a) Let σ be a uniformly distributed random element of $T_n(m!\,(n-m))$. Then, with probability greater than $1-\varepsilon$, σ contains a cycle of length $n-m$.

(b) Let $2 \le s \le m$, and let σ be a uniformly distributed random element of $T_n((n-m)s)$. Then, with probability greater than $(1-\varepsilon)/m!$, the cycle structure of σ is $1^{m-s}s^1(n-m)^1$.

Proof. (a) This is immediate from Lemma 3.6 and Theorem 3.7.

(b) $T_n((n-m)s) \subseteq T_n(m!\,(n-m))$, so it has at most $(1+\varepsilon)n!/n$ elements. Out of these, there are $n!/s(n-m)(m-s)! > n!/m!\,n$ with cycle structure $1^{m-s}s^1(n-m)^1$, so the proportion of elements in $T_n((n-m)s)$ with the required cycle structure is greater than $(1-\varepsilon)/m!$. □

Corollary 3.8 covers all cases occurring in Lemma 3.5, since if $m \in \{0,1\}$ then part (a) of the corollary can also be interpreted as a conditional probability of permutations with the required cycle structure. Note that if x is odd then $T_n(x) \subseteq A_n$, so if $(n-m)s$ is odd then the conditional probability that an element $\sigma \in A_n$ has cycle structure $1^{m-s}s^1(n-m)^1$, given that $\sigma^{(n-m)s} = 1$, is the same as the corresponding conditional probability in S_n. However, for our algorithmic application, we need a lower bound for the conditional probability which is valid for all values of n.

Theorem 3.9. *Let $n \ge 5$ and let σ be a randomly selected permutation from S_n. Let $x = n$ or $x = (n-m)s$ for one of the cycle types $1^{m-s}s^1(n-m)^1$ described in Lemma 3.5. Then, given that $\sigma^x = 1$, the conditional probability that σ is an n-cycle, or σ has cycle structure $1^{m-s}s^1(n-m)^1$, is at least $1/180$.*

Proof. Using the notation of the proof of Theorem 3.7, we derive a tighter upper bound for $N_n(x)$ by evaluating (3.1) more carefully in the case $k = 3$. First, we suppose that $n > 50$.

There is one partition Π_3 of $\{1,2,3\}$ with three parts, and (3.1) gives $N(3,\Pi_3,d_1,d_2,d_3) \le (n-3)!\,d_1^0 d_2^0 d_3^0 = (n-3)!$. By Lemmas 3.1 and 3.2,

$$N(3,\Pi_3) \le c_3\,(n-m)\,(n-3)! \le c_3\,n\,(n-3)!$$

with $c_3 = c_2^3$ for the constant c_2 introduced in Lemma 3.2.

There are three partitions Π_2 of $\{1,2,3\}$ with two parts, and (3.1) gives $N(3,\Pi_2,d_1,d_2) \le (n-3)!\,d_1 d_2^0$. Hence, using both statements of Lemmas 3.1 and 3.2,

$$N(3,\Pi_2) \le (n-3)! c_2 n^{4/3}(c_4 + 2\ln n)$$

with $c_4 = 3 - 4\ln 3$. In this estimate, we used Lemma 3.2 with $p = 3$, since for $n > 50$ this value gives the largest upper bound.

Finally, there is one partition Π_1 of $\{1,2,3\}$ with one part, $N(\Pi_1, d_1) \le (n-3)!\, d_1^2$ and, by Lemmas 3.3 and 3.4, $N(3,\Pi_1) \le 4n^2\,(n-3)!$. (Again, the case $y=3$ yields the largest upper estimate.) Adding these estimates, we obtain

$$N_n(x) \le (n-3)!\Big(4n^2 + 3c_2 n^{4/3}(c_4 + 2\ln n) + c_3 n\Big) = f(n)(n-1)!.$$

The function $f(n)$ is monotone decreasing for $n \ge 50$.

We claim that $N_n(x) \le 10(n-1)!$ for all $n \ge 5$. For $n \ge 301$ this follows from $f(301) < 10$, and for $5 \le n \le 300$, we computed the exact value of $N_n(x)$.

By Lemma 3.5 and by checking the cases $n \le 9$ which are not covered by that lemma, the number of permutations with the required cycle structure is at least $(n-1)!/18$. Hence the proportion of these elements in $T_n(x)$ is at least $1/180$. □

4. Construction of the new generators

Given a black-box group $G = \langle S\rangle$ isomorphic to A_n or S_n, in this section we describe an algorithm which constructs $s, t \in G$ such that the subgroup $\langle s,t\rangle$ satisfies the presentation (2.2) or (2.3), if n is odd or even, respectively.

We construct random elements of G (we shall compute in the proof of Theorem 4.5 how many of them have to be taken) in order to find $a \in G$ satisfying $a^{n-k} = 1$, where $k = 0$ if n is odd, and $k = 1$ if n is even. Also, we construct random elements $c \in G$ in order to find one satisfying $c^{3(n-m)} = 1$, where m is given in the following table.

(4.1)

$n \mod 6$	0	1	2	3	4	5
m	5	6	3	4	3	4

Then, by Theorem 3.9, the cycle type of a is $1^k(n-k)^1$ with probability at least $1/180$, and the cycle type of $b := c^{n-m}$ is $1^{n-3}3^1$ with probability at least $1/180$.

The following two lemmas describe algorithms which, given $a, b \in G$ as above, construct $s, t \in G$ as required in (2.2) or (2.3). These algorithms are Las Vegas, which means that if the input group elements a, b have cycle type $1^k(n-k)^1$ and $1^{n-3}3^1$, respectively, then the output group elements s, t behave as required, or the algorithm reports failure. However, if the input elements do not have the prescribed cycle types then the algorithms may return an incorrect answer. If the output group elements s, t are incorrect then this fact is noticed when we check whether $\langle s,t\rangle$ satisfies the presentation (2.2) or (2.3).

Lemma 4.1. *Let n be odd and $n \ge 7$. Suppose that $a \in G$ has cycle type n^1 and $b \in G$ has cycle type $1^{n-3}3^1$. Then in $O(\xi n + \mu n)$ time, it is possible to construct $s, t \in G$ such that $\langle s,t\rangle$ satisfies the presentation (2.2) for A_n. This algorithm is Las Vegas and succeeds with probability at least $3/4$.*

Proof. Let us fix a homomorphism $\lambda : G \to \mathrm{Sym}([1,n])$ such that $\lambda(a) = (1,2,\ldots,n)$. Our first goal is to show that, with probability at least $3/4$, among $1 + n/3$ random conjugates of b we find one, c, satisfying $\lambda(c) = (i, i+1, k)$ or $\lambda(c) = (i+1, i, k)$ with $1 \le i, k \le n$ and $k \notin \{i, i+1\}$, where the numbers are taken modulo n.

Suppose c is a conjugate of b satisfying $[c, c^a] \ne 1$. We claim that $\lambda(c)$ has support set $\{i, i+1, k\}$ as desired. Indeed, as c is a conjugate of b, it satisfies $\lambda(c) = (i,j,k)$ for some triple $\{i,j,k\}$ and $\lambda(c^a) = (i+1, j+1, k+1)$. Now c and

c^a do not commute if and only if the sets $\{i,j,k\}$ and $\{i+1,j+1,k+1\}$ intersect. Hence it follows that two of i,j,k are consecutive numbers modulo n.

Next we show that the probability that we cannot find such an element c among $1+n/3$ random conjugates of b is less than $1/4$. There are $\binom{n}{3}$ possible support sets for a 3-cycle, and out of these, $n(n-3)$ contain two consecutive numbers modulo n. Hence one random conjugate succeeds with probability $n(n-3)/\binom{n}{3} = 6(n-3)/((n-1)(n-2))$, and the probability that $1+n/3 > 2(n-1)(n-2)/(6(n-3))$ random conjugates succeed is greater than

$$1-(1-\frac{6(n-3)}{(n-1)(n-2)})^{\frac{2(n-1)(n-2)}{6(n-3)}} > 1-\frac{1}{e^2} > \frac{3}{4}.$$

The rest of the algorithm is deterministic and runs in $O(\mu)$ time. Without loss of generality, we can suppose that $\mathrm{supp}(\lambda(c)) = \{1,2,k\}$ for some k with $3 \le k \le n-1$. The next goal is to construct $t \in G$ such that $\lambda(t) = (1,2,3)$.

If $k=3$ then cc^a is an involution while if $4 \le k \le n-1$ then cc^a has order 5. Hence these two cases can be distinguished in $O(\mu)$ time.

Suppose first that $k=3$. We can distinguish the cases $\lambda(c) = (1,2,3)$ and $\lambda(c) = (1,3,2)$ by computing $x := c^{a^2}$ and $y := c^x$. In the first case $\lambda(y) = (1,2,4)$ and in the second case $\lambda(y) = (1,5,2)$, which can be distinguished by checking whether $[y, y^{a^2}] = 1$. After that, t is defined as $t := c$ or $t := c^2$, respectively.

Suppose next that $4 \le k \le n-1$. The case $k \in \{4, n-1\}$ can be distinguished from the case $5 \le k \le n-2$ by checking whether $[c, c^{a^2}] = 1$. If $5 \le k \le n-2$ then $\lambda(c) = (1,2,k)$ can be distinguished from $\lambda(c) = (1,k,2)$ by computing $x := c^a$ and $y := c^x$. In the first case $\lambda(y) = (1,3,k)$ and in the second case $\lambda(y) = (1,k,k+1)$, which can be distinguished by checking whether $[y, y^a] = 1$. If it turns out that $\lambda(c) = (1,2,k)$ then define $t := [c^2, x]$. If $\lambda(c) = (1,k,2)$ then define $t := [c, x^2]$.

If $k \in \{4, n-1\}$ then the cases $\lambda(c) = (1,2,4)$, $\lambda(c) = (1,4,2)$, $\lambda(c) = (1,2,n-1)$, and $\lambda(c) = (2,1,n-1)$ are also distinguished by computing $x := c^a$ and $y := c^x$. In the four cases, $\lambda(y) = (1,3,4)$, $\lambda(y) = (1,4,5)$, $\lambda(y) = (n-1,1,3)$, and $\lambda(y) = (n-1,n,1)$, respectively. The third of these is distinguished from the others as the only one with $[y, y^a] = 1$. The second one is distinguished among the remaining three as the only one with $[y, y^{a^2}] = 1$. Finally, the first and fourth are distinguished by the order of yy^a. If $\lambda(c) = (1,2,4)$ or $\lambda(c) = (1,2,n-1)$ then define $t := [c^2, x]$. If $\lambda(c) = (1,4,2)$ or $\lambda(c) = (2,1,n-1)$ then define $t := [c, x^2]$.

Finally, output $s := at^2$, which satisfies $\lambda(s) = (3,4,\ldots,n)$, and t. □

Remark 4.2. For use in [4], we point out the following corollary of the proof of Lemma 4.1. Given $a, c \in G$ such that $\lambda(a) = (1,2,\ldots,n)$ and $\lambda(c) \in \{(i,i+1,k),(i+1,i,k)\}$ for some $i,k \le n$, we can construct $s,t \in G$ satisfying the presentation (2.2) for A_n in $O(\mu)$ time, by a deterministic algorithm.

Lemma 4.3. *Let n be even and $n \ge 10$. Suppose that $a \in G$ has cycle type $1^1(n-1)^1$ and $b \in G$ has cycle type $1^{n-3}3^1$. Then in $O(\xi n + \mu n)$ time, it is possible to construct $s,t \in G$ such that $\langle s,t\rangle$ satisfies the presentation (2.3) for A_n. This algorithm is Las Vegas and succeeds with probability at least $3/4$.*

Proof. Let us fix a homomorphism $\lambda : G \to \mathrm{Sym}([1,n])$ such that $\lambda(a) = (2,3,\ldots,n)$. Our first goal is to show that, with probability at least $3/4$, among $2n/3$ random conjugates of b we find one, c, satisfying $\lambda(c) = (1,i,j)$ with $2 \le i,j \le n$.

Suppose c is a conjugate of b satisfying $[c, c^a] \neq 1$, $[c, c^{a^2}] \neq 1$, and $[c, c^{a^4}] \neq 1$. We claim that $\lambda(c)$ has support set $\{1, i, j\}$ as desired. Indeed, as c is a conjugate of b, if $1 \in \mathrm{supp}(\lambda(c))$ then c does not commute with c^{a^m} for any m. On the other hand, if $\mathrm{supp}(\lambda(c)) = \{i, j, k\} \subseteq \{2, 3, \ldots, n\}$ then c commutes with c^a if no two of i, j, k are consecutive in the $(n-1)$-cycle $\lambda(a)$ and in the other cases, it is easy to check that c commutes with c^{a^2} or c^{a^4}.

Next we show that the probability that we cannot find such an element c among $2n/3$ random conjugates of b is less than $1/4$. There are $\binom{n}{3}$ possible support sets for a 3-cycle, and out of these, $\binom{n-1}{2}$ contain 1. One random conjugate succeeds with probability $\binom{n-1}{2}/\binom{n}{3} = 3/n$. Hence the probability that $2n/3$ random conjugates succeed is greater than $1 - (1-3/n)^{2n/3} > 1 - 1/e^2 > 3/4$.

Define $t := [c^a, c]$. Then $\lambda(t) = (1, i, i+1)$ and, without loss of generality, we may suppose $\lambda(t) = (1, 2, 3)$. Then $s := at$ satisfies $\lambda(s) = (1, 2)(3, 4, \ldots, n)$. Output s and t. □

Lemma 4.4. *Given $s, t \in G$, it can be checked in $O(\mu n)$ time whether $\langle s, t\rangle$ satisfies the presentation for A_n given in (2.2) and (2.3).*

Proof. The case n even, as well as the evaluation of the relators s^{n-2}, t^3, and $(st)^n$ in the odd case is clear. In the case when n is odd, we evaluate the relators $(ts^{-k}ts^k)^2$ for $k = 1, \ldots, \lfloor (n-3)/2 \rfloor$ in $\lfloor (n-3)/2 \rfloor$ rounds: in the k^{th} round, we use the input s^{k-1} and we output s^k and $(ts^{-k}ts^k)^2$. One round requires only a constant number of group operations. □

Theorem 4.5. *Given a black-box group $G = \langle S\rangle$ isomorphic to A_n or S_n and an error probability $\varepsilon > 0$, group elements $s, t \in G$ such that $\langle s, t\rangle$ satisfies the presentation for A_n given in (2.2) or (2.3) can be constructed by a Las Vegas algorithm, with probability at least $1 - \varepsilon$, in $O(\log \varepsilon^{-1}(\xi n + \mu n \log n))$ time. At any moment during the execution of the algorithm, we have to store only a constant number of group elements.*

Proof. By Lemma 3.5, among $2n$ uniformly distributed random elements $a \in G$ we can find one satisfying $a^{n-k} = 1$, with the appropriate $k \in \{0, 1\}$, with probability at least $1-(1-1/n)^{2n} > 1-1/e^2 > 3/4$. Similarly, among $36n$ uniformly distributed random elements $c \in G$ we can find one satisfying $c^{3(n-m)} = 1$ for the value m described in (4.1), with probability greater than $3/4$. Constructing a, c and taking the appropriate powers can be done in $O(\xi n + \mu n \log n)$ time. By Theorem 3.9, a has cycle type $1^k(n-k)^1$ and c^{n-m} has cycle type $1^{n-3}3^1$ with probability at least $1/180^2$. Applying the appropriate one of Lemmas 4.1 and 4.3, and then Lemma 4.4, if a and c have the correct cycle type then s and t are constructed in $O(\xi n + \mu n)$ time, with probability at least $3/4$. Hence the entire procedure takes $O(\xi n + \mu n \log n)$ time and succeeds with probability greater than $(3/4)^3/180^2$.

Repeating this procedure up to $\lceil (4/3)^3 180^2 \ln(\varepsilon^{-1})\rceil$ times, we construct s and t with probability at least $1 - \varepsilon$, in $O(\log \varepsilon^{-1}(\xi n + \mu n \log n))$ time. □

Remark 4.6. For the "practical" values $8 \leq n \leq 300$, we have computed the exact value of the conditional probabilities that an element of order dividing $s(n-m)$ has cycle type $1^{n-m-s}s^1(n-m)^1$ for the values s, m in Lemma 3.5. All of these conditional probabilities are greater than $1/7$ so, for $8 \leq n \leq 300$, the number $\lceil (4/3)^3 180^2 \ln(\varepsilon^{-1})\rceil$ can be replaced by $\lceil (4/3)^3 7^2 \ln(\varepsilon^{-1})\rceil$ in the expression for the

number of iterations in the proof of Theorem 4.5. This reduction does not really matter if the input group is indeed isomorphic to A_n or S_n, since the expected number of iterations depends on the true conditional probabilities that an element of order dividing $(n-m)s$ has cycle type $1^{n-m-s}s^1(n-m)^1$ for the appropriate values of s and m, and it does not depend on how badly or well we estimate these conditional probabilities. However, if the algorithm is used as a Monte Carlo algorithm to test whether an unknown input group G is isomorphic to A_n or S_n then the better constants ensure earlier termination in the case of a negative answer.

5. The homomorphism λ

Given a black-box group $G = \langle S \rangle$ isomorphic to A_n or S_n, the algorithm described in the proof of Theorem 4.5 constructs $s, t \in G$ such that $\langle s, t \rangle \cong A_n$ and it satisfies the presentation in (2.2) or (2.3). In this section we construct a homomorphism $\lambda : G \to \mathrm{Sym}([1,n])$ by specifying the images of s and t and by giving a procedure which constructs the image of any $z \in G$. The algorithm will detect if $G \cong S_n$, and in this case it replaces s and t by two new elements s_1, t_1 such that $\langle s_1, t_1 \rangle$ satisfies (2.1). We shall also describe a procedure which, given an arbitrary element $z \in G$, computes a straight-line program reaching z from s and t (or from s_1 and t_1 in the case $G \cong S_n$).

Recall that for $n > 6$ we have $\mathrm{Aut}(A_n) \cong S_n$, and so, for any $g \in G$, the cycle structure of $\lambda(g)$ is the same for any faithful homomorphism $\lambda : G \to \mathrm{Sym}([1,n])$. Therefore, without loss of generality, we can assume that $\lambda(s) = (3,4,\ldots,n)$ or $\lambda(s) = (1,2)(3,4,\ldots,n)$, depending on the parity of n, and $\lambda(t) = (1,2,3)$.

We start with the easier inverse problem: finding straight-line programs reaching any $\pi \in A_n$ from $\sigma := \lambda(s)$ and $\tau := \lambda(t)$.

Lemma 5.1. *Given $\pi \in A_n$, a straight-line program of length $O(n \log n)$ reaching π from σ and τ can be constructed in $O(n \log n)$ time, by a deterministic algorithm.*

Proof. Let $\sigma_1 := \sigma$ and $\tau_1 := \tau$. Recursively for $i = 1, \ldots, n-3$, define

$$\tau_{i+1} = \begin{cases} \tau_i^{-1}\sigma_i^{-1}\tau_i\sigma_i\tau_i & \text{if } n-i \text{ is even} \\ \tau_i^{-1}\sigma_i^{-1}\tau_i^2\sigma_i\tau_i & \text{if } n-i \text{ is odd,} \end{cases}$$

$$\sigma_{i+1} = \begin{cases} \sigma_i\tau_{i+1} & \text{if } n-i \text{ is even} \\ \sigma_i\tau_i^2\tau_{i+1}^{-1} & \text{if } n-i \text{ is odd.} \end{cases} \tag{5.1}$$

Then $\tau_i = (i, i+1, i+2)$ for $1 \le i \le n-2$ and $\sigma_i = (i, i+1)(i+2, i+3, \ldots, n)$ or $(i+2, i+3, \ldots, n)$, depending on the parity of $n-i$. It is clear from the recursion that all σ_i, τ_i can be obtained by a single straight-line program of length $O(n)$ from σ and τ. Hence it is enough to write a straight-line program of length $O(n \log n)$ from $T := \{\sigma_i, \tau_i : 1 \le i \le n-2\}$ to π.

Write π as the product of transpositions, by decomposing each cycle $(c_1, \ldots, c_\ell)$ of π as $(c_1, \ldots, c_\ell) = (c_1, c_2)(c_1, c_3)\cdots(c_1, c_\ell)$. Since $\pi \in A_n$, we have an even number of transpositions in this product. By inserting $(n-1, n)(n-1, n)$ between the $(2k-1)^{\text{st}}$ and $(2k)^{\text{th}}$ transposition for all k, the permutation π is written as the product of less than n permutations of the form $(i,j)(n-1,n)$ or $(n-1,n)(i,j)$ and it is enough to show that any such permutation can be obtained by a straight-line program of length $O(\log n)$ from T.

For any $k \in [i+2, n]$, we have

$$\sigma_i^{-(k-i-2)}\tau_i\sigma_i^{k-i-2} = (i, i+1, k) \quad \text{or} \quad \sigma_i^{-(k-i-2)}\tau_i\sigma_i^{k-i-2} = (i+1, i, k),$$

and these 3-cycles can be reached by straight-line programs of length $O(\log n)$ from T. Hence it is enough to observe that if $i < n-1$, then

$$(i,i+1)(n-1,n) = (i,n,i+1)(i,i+1,n-1)(i,n,i+1)$$

and, for $j \in [i+2, n-2]$, $(i,j)(n-1,n) = (i,i+1,j)\cdot(i,i+1)(n-1,n)\cdot(i,j,i+1)$. If $i < n-2$ and $j \in \{n-1,n\}$, then $(i,j)(n-1,n), (n-1,n)(i,j) \in \langle (i,n-1,n)\rangle$ and $(i,n-1,n) = (i,n-1,i+1)(i,i+1,n)$. □

The evaluation of a straight-line program of length $O(n\log n)$ may require the simultaneous storage of $cn\log n$ group elements, but in the case of the straight-line programs constructed in Lemma 5.1, we can do much better.

Corollary 5.2. *Given $\pi \in A_n$, the inverse image $\lambda^{-1}(\pi) \in G$ can be computed in $O(\mu n \log n)$ time, by a deterministic algorithm. At any moment during the execution of this algorithm, we have to store only a constant number of elements of G.*

Proof. We simply evaluate in G the straight-line program reaching π from σ and τ, starting with s and t. The storage requirement can be satisfied by ordering the cycles of π according to the smallest element contained in them and decomposing each cycle as $(c_1,\dots,c_\ell) = (c_1,c_2)(c_1,c_3)\cdots(c_1,c_\ell)$ using its smallest element c_1. Then transpositions (i,j) with $i<j$ for some fixed i occur consecutively, and all later transpositions (i_1,j_1) have $i_1,j_1 > i$. Therefore, evaluating the straight-line programs reaching the preimages of the elements $(i,j)(n-1,n)$ and $(n-1,n)(i,j)$ successively as required in the proof of Lemma 5.1, we have to store $\lambda^{-1}(\sigma_i)$ and $\lambda^{-1}(\tau_i)$ only for a fixed i at any given time. Note that an inverse image $\lambda^{-1}(\sigma_i^{k-i-2})$ can be computed by repeated squaring, storing only a constant number of elements of G at any time. After processing all (i,j) with this fixed i, we compute $\lambda^{-1}(\sigma_{i+1})$ and $\lambda^{-1}(\tau_{i+1})$ by the formulas in (5.1) and discard $\lambda^{-1}(\sigma_i)$ and $\lambda^{-1}(\tau_i)$. □

The main idea of the construction of $\lambda(z)$ for an arbitrary $z \in G$ is the following. We need to define an n-element set Ω on which G acts, and then we need to identify Ω with $\{1,2,\dots,n\}$. We define Ω as a set of unordered pairs $\{a,b\} \subseteq G$ such that both a and b have cycle type $1^{n-3}3^1$, satisfying the requirement that the supports of $\lambda(a)$ and $\lambda(b)$ intersect in exactly one point $i \in [1,n]$. In this way, $\{a,b\}$ can be identified with i, and Ω can be identified with $[1,n]$.

Representing Ω in this way creates two problems. First, we do not want to store $2n$ elements of G, so the set Ω is not computed explicitly. Elements of Ω, when needed, will be reached by straight-line programs from s and t. Second, for an arbitrary $z \in G$ the image $\{a^z, b^z\}$ of a point $\{a,b\} \in \Omega$ is not necessarily an element of Ω. Thus we need to be able to identify the intersection of the supports of $\lambda(a^z)$ and $\lambda(b^z)$ with $\mathrm{supp}(\lambda(c)) \cap \mathrm{supp}(\lambda(d))$ for some $\{c,d\} \in \Omega$ in a way different from simply checking whether $\{a^z,b^z\} = \{c,d\}$. We shall narrow down the possibilities for $\mathrm{supp}(\lambda(a^z)) \cap \mathrm{supp}(\lambda(b^z))$ to a constant number by taking commutators of a^z and b^z with certain elements $x_1,\dots,x_m$ of G of order five, utilizing the fact that an element of order five and a three-cycle in S_n commute if and only if their supports are disjoint.

Now we describe the construction of the elements $x_1,\dots,x_m$. Let $n = 5k+r$ where $0 \le r \le 4$, let $m' := \lceil \log_2(k+1)\rceil$, and put $m := 2m'$. For $1 \le j \le m'$, we define partitions $\Pi_j = \{P_{j,0}, P_{j,1}\}$ of the set $\{1,\dots,k\}$ into two parts such that the common refinement of these partitions is the trivial one. Namely, for each $i \in [1,k]$,

we compute the binary expansion $b_1^{(i)}b_2^{(i)}\dots b_{m'}^{(i)}$ of i. For $i \in [1,k]$ and $j \in [1,m]$, let

$$\varepsilon(j,i) = \begin{cases} b_j^{(i)} & \text{if } j \le m' \\ 1 - b_{j-m'}^{(i)} & \text{if } j \ge m'+1. \end{cases} \tag{5.2}$$

Then, for $1 \le j \le m'$, we define $P_{j,0} := \{i : \varepsilon(j,i) = 0\}$ and $P_{j,1} := \{i : \varepsilon(j,i) = 1\}$.

We define elements $a_1, \dots, a_k$ of G with cycle type $1^{n-5}5^1$ as follows. Working in the group $A := \langle \{s^{-e}ts^e : 0 \le e \le 7\}\rangle \cong A_{10}$, in $O(\mu)$ time we construct $a_1, a_2 \in \langle s,t\rangle$ with $\lambda(a_1) = (1,2,3,4,5)$ and $\lambda(a_2) = (6,7,8,9,10)$. We also compute $c := s^5$ and *define, but do not compute*, $a_i := c^{-(i-2)}a_2c^{i-2}$ for $3 \le i \le k$. Note that $\lambda(a_i) = (5i-4, 5i-3, 5i-2, 5i-1, 5i)$.

Finally we define $x_j := \prod_{i=1}^k a_i^{\varepsilon(j,i)}$ for $1 \le j \le m$. Thus each x_j has order five. For $j \le m'$, we have $\{i : \mathrm{supp}(\lambda(a_i)) \subseteq \mathrm{supp}(\lambda(x_j))\} = P_{j,1}$. Similarly, for $j > m'$, we have $\{i : \mathrm{supp}(\lambda(a_i)) \subseteq \mathrm{supp}(\lambda(x_j))\} = P_{j-m',0}$ and so the supports of $\lambda(x_j)$ and $\lambda(x_{j+m'})$ are disjoint for all $j \in [1,m']$.

Note that for any $J \subseteq \{1,2,\dots,m'\}$,

$$\left| \bigcap_{j\in J} \mathrm{supp}(\lambda(x_j)) \cap \bigcap_{j\in\{1,2,\dots,m'\}\setminus J} \mathrm{supp}(\lambda(x_{j+m'})) \right| \in \{0,5\}. \tag{5.3}$$

If this intersection is non-empty, then it is the support of $\lambda(a_i)$ for the unique i whose binary expansion contains 1's exactly in the positions given in J.

Lemma 5.3. *Given a_1, a_2, and c as defined above, $x_1, \dots, x_m$ and their images under λ can be computed in $O(\mu n \log n)$ time, storing only $O(\log n)$ elements of G at any moment of the computation.*

Proof. Initialize $x_j := 1$ for $1 \le j \le m$. Iteratively for $i = 1, \dots, k$, compute $x_j := x_j a_i^{\varepsilon(j,i)}$ for all j, compute $a_{i+1} = a_i^c$, and discard a_i. □

Lemma 5.4. *Given $z \in G$ and $\{x_1, \dots, x_m\}$ as constructed above, for any $\ell \in \{1, \dots, n\}$ the image $\ell^{\lambda(z)}$ can be determined in $O(\mu \log n)$ time, by a deterministic algorithm.*

Proof. We choose some i with $1 \le i \le n-4$ such that $\ell \in \{i, i+1, i+2, i+3, i+4\}$, and determine $i^{\lambda(z)}, (i+1)^{\lambda(z)}, (i+2)^{\lambda(z)}, (i+3)^{\lambda(z)}$, and $(i+4)^{\lambda(z)}$ simultaneously. First, we construct the ten elements $t_{\{i_1,i_2,i_3\}}$ of G with the property that $\mathrm{supp}(\lambda(t_{\{i_1,i_2,i_3\}})) = \{i_1,i_2,i_3\}$, for all three-element subsets $\{i_1,i_2,i_3\} \subseteq \{i, i+1, i+2, i+3, i+4\}$. This can be done by constructing these ten elements $t_{\{i_1,i_2,i_3\}}$ for $6 \le i_1, i_2, i_3 \le 10$ in the group $A := \langle \{s^{-e}ts^e : 0 \le e \le 7\}\rangle \cong A_{10}$ in $O(\mu)$ time and, for $i \ge 3$, conjugating the result by s^{i-6}. For $i = 1, 2$, we construct those elements directly in A.

Let $j \in [1,m']$ be fixed. We construct the twenty commutators $[x_j, (t_{\{i_1,i_2,i_3\}})^z]$, $[x_{j+m'}, (t_{\{i_1,i_2,i_3\}})^z]$. At least one of these twenty commutators is trivial, since at least three elements of $\{i, i+1, i+2, i+3, i+4\}^{\lambda(z)}$ are outside $\mathrm{supp}(\lambda(x_j))$ or $\mathrm{supp}(\lambda(x_{j+m'}))$, and two permutations with disjoint support commute.

Suppose, for example, that $[x_j, (t_{\{i,i+1,i+2\}})^z] = 1$; in the other nineteen cases, we can use an analogous argument. The fact $[x_j, (t_{\{i,i+1,i+2\}})^z] = 1$ means that

$$i^{\lambda(z)}, (i+1)^{\lambda(z)}, (i+2)^{\lambda(z)} \in \mathrm{supp}(\lambda(x_{j+m'})) \cup \{5k+1, \dots, 5k+r\}.$$

We also want to compute which of the sets $\mathrm{supp}(\lambda(x_j)) \cup \{5k+1,\dots,5k+r\}$ and $\mathrm{supp}(\lambda(x_{j+m'})) \cup \{5k+1,\dots,5k+r\}$ contain the other two images $(i+3)^{\lambda(z)}$ and $(i+4)^{\lambda(z)}$. If $[x_j,(t_{\{i,i+1,i+3\}})^z] \neq 1$ then $(i+3)^{\lambda(z)} \in \mathrm{supp}(\lambda(x_j)) \cup \{5k+1,\dots,5k+r\}$, since $\{i,i+1,i+3\}^{\lambda(z)}$ can intersect $\mathrm{supp}(\lambda(x_j))$ only in the point $(i+3)^{\lambda(z)}$. If $[x_j,(t_{\{i,i+1,i+3\}})^z] = 1$ then $(i+3)^{\lambda(z)} \in \mathrm{supp}(\lambda(x_{j+m'})) \cup \{5k+1,\dots,5k+r\}$. We can decide similarly which of these two sets contains $(i+4)^{\lambda(z)}$.

Having performed the commutator calculations of the previous two paragraphs for all $j \in [1,m']$, by (5.3) we have at most $5+r \leq 9$ possibilities for $\ell^{\lambda(z)}$. To finish the algorithm, we need a procedure which decides whether $\ell^{\lambda(z)} = \bar{\imath}$ for a fixed $\bar{\imath} \in [1,n]$, in $O(\mu \log n)$ time.

We construct $s_1,s_2,s_3,s_4,s_5 \in \langle s,t\rangle$ of cycle type $1^{n-3}3^1$, such that $\mathrm{supp}(\lambda(s_{d_1})) \cap \mathrm{supp}(\lambda(s_{d_2})) = \{\bar{\imath}\}$ for any two distinct $d_1,d_2 \in [1,5]$. Again, these s_d can be obtained as conjugates of appropriate elements of A. We also pick $t_{\{i_1,i_2,i_3\}}$ and $t_{\{i'_1,i'_2,i'_3\}}$ from our collection of ten group elements such that $\{i_1,i_2,i_3\} \cap \{i'_1,i'_2,i'_3\} = \{\ell\}$. We claim that $\ell^{\lambda(z)} = \bar{\imath}$ if and only if $[s_d,(t_{\{i_1,i_2,i_3\}})^z] \neq 1$ for at least four $d \in [1,5]$ and $[s_d,(t_{\{i'_1,i'_2,i'_3\}})^z] \neq 1$ for at least four $d \in [1,5]$. Indeed, if $\ell^{\lambda(z)} = \bar{\imath}$ then $\mathrm{supp}(\lambda((t_{\{i_1,i_2,i_3\}})^z))$ intersects but is not equal to at least four of the sets $\mathrm{supp}(\lambda(s_d))$ and so $(t_{\{i'_1,i'_2,i'_3\}})^z$ and s_d do not commute. The same argument works for $(t_{\{i'_1,i'_2,i'_3\}})^z$ as well. Conversely, if $[s_d,(t_{\{i_1,i_2,i_3\}})^z] \neq 1$ for at least four $d \in [1,5]$ then $\mathrm{supp}(\lambda((t_{\{i_1,i_2,i_3\}})^z))$ intersects four three-element sets with pairwise intersection $\bar{\imath}$. This can happen only if $\bar{\imath} \in \{i_1,i_2,i_3\}^{\lambda(z)}$ and similarly $\bar{\imath} \in \{i'_1,i'_2,i'_3\}^{\lambda(z)}$. □

Lemma 5.5. (*a*) *Given s,t and $x_1,\dots,x_m$, it can be decided in $O(\mu|X|n\log n)$ time whether $G \cong A_n$ or $G \cong S_n$.*

(*b*) *If it turns out that $G \cong S_n$ then generators s_1,t_1 for G satisfying (2.1) can be computed in $O(\mu n \log n)$ time.*

(*c*) *If $G \cong S_n$ then for any permutation $\sigma \in S_n$, a straight-line program of length $O(n\log n)$ reaching σ from $\lambda(s_1)$ and $\lambda(t_1)$ can be written by a deterministic algorithm, in $O(n\log n)$ time. The inverse image $\lambda^{-1}(\sigma)$ can be computed in $O(\mu n\log n)$ time, storing only a constant number of elements of G at any moment during the computation.*

Proof. (a) By Lemma 5.4, the images $\lambda(z)$ of all input generators $z \in X$ can be computed in $O(\mu|X|n\log n)$ time. All $\lambda(z)$ are even permutations if and only if $G \cong A_n$.

(b) Suppose that we have found $z_0 \in X$ such that $\pi := \lambda(z_0)$ is an odd permutation. By Corollary 5.2, $z_1 := \lambda^{-1}(\pi\cdot(1,2))$ can be computed in $O(\mu n\log n)$ time, and then $t_1 := z_0^{-1}z_1$ satisfies $\lambda(t_1) = (1,2)$. Depending on the parity of n, we compute $z_2 := s$ or $z_2 := t_1 s$ such that $\lambda(z_2) = (3,4,\dots,n)$. Finally, we compute $s_1 := z_2 t$, which satisfies $\lambda(s_1) = (1,2,\dots,n)$.

(c) Depending on the parity of σ, we compute a straight-line program reaching σ or $\sigma\cdot(1,2)$ from $\lambda(s)$ and $\lambda(t)$, as described in Lemma 5.1. Then, it is enough to observe that $\lambda(s)$ and $\lambda(t)$ can be reached from $\lambda(s_1)$ and $\lambda(t_1)$ by a straight-line program of constant length, since $t = [s_1,t_1]$ and $s = s_1t^2$ if n is odd, and $s = t_1s_1t^2$ if n is even.

The evaluation of this straight-line program in G is done as described in Corollary 5.2. □

Lemma 5.6. *Given any $z \in G$, a straight-line program of length $O(n \log n)$ and reaching z from s_1, t_1 in the case $G \cong S_n$ and from s, t in the case $G \cong A_n$ can be computed in $O(\mu n \log n)$ time.*

Proof. Using the algorithm described in the proof of Lemma 5.4, we compute $\lambda(z)$. By Lemma 5.1 and Lemma 5.5(c), we can write a straight-line program reaching $\lambda(z)$ from $\lambda(s), \lambda(t)$ or $\lambda(s_1), \lambda(t_1)$, respectively. The same straight-line program reaches z from s, t or s_1, t_1. □

Summary of the proof of Theorem 1.2(a). The decision procedure whether $G \cong A_n$ or $G \cong S_n$ is described in Lemma 5.5(a), and the new generators for G are constructed in the proof of Theorem 4.5 for $G \cong A_n$ and in the proof of Lemma 5.5(b) for $G \cong S_n$. Given $g \in G$, the construction of $\lambda(g)$ is described in Lemma 5.4, and the straight-line program reaching g is constructed in Lemma 5.6. Finally, the inverse image of a permutation is constructed in Lemma 5.1 and Corollary 5.2 in the case $G \cong A_n$ and in Lemma 5.5(c) for $G \cong S_n$.

Proof of Theorem 1.2(b). Given $G = \langle X \rangle$ of unknown isomorphism type, we attempt to construct $s, t, x_1, \ldots, x_m$ (and s_1, t_1, if necessary). If the construction fails then with large probability G is not isomorphic to A_n or S_n. If the construction succeeds and s_1, t_1 have been computed then we check that $\langle s_1, t_1 \rangle$ satisfies (2.1) (recall that it has been checked during the construction that $\langle s, t \rangle$ satisfies the appropriate one of (2.2) or (2.3)).

Finally, we write straight-line programs from s, t or s_1, t_1 to each generator $z \in X$, as described in the proof of Lemma 5.6, and evaluate the straight-line programs. If the construction of the straight-line program succeeds and the evaluated value is equal to z for all $z \in X$ then we know with certainty that G is A_n or S_n. If not, then G is not isomorphic to A_n or S_n.

Acknowledgment

We thank Colva Roney-Dougal, whose careful reading of the manuscript led to the correction of several minor inaccuracies.

References

[1] László Babai, *Local expansion of vertex-transitive graphs and random generation in finite groups*, Proc. 23rd ACM Sympos. Theory Comp. 1991, pp. 164–174.

[2] László Babai and Endre Szemerédi, *On the complexity of matrix group problems, I*, Proc. 25th IEEE Sympos. Foundations Comp. Sci., 1984, pp. 229–240.

[3] Robert Beals and László Babai, *Las Vegas algorithms for matrix groups*, Proc. 34rd IEEE Sympos. Foundations Comp. Sci., 1993, pp. 427–436.CMP 95:11

[4] Robert Beals, Charles Leedham-Green, Alice Niemeyer, Cheryl Praeger, and Ákos Seress, *A black-box algorithm for recognizing finite symmetric and alternating groups, II*, (2002), preprint.

[5] ———, *Constructive recognition of finite alternating and symmetric groups acting as matrix groups on their natural permutation modules*, (2002), preprint.

[6] Sergei Bratus, *Recognition of finite black box groups*, Ph.D. thesis, Northeastern University, 1999.

[7] Sergey Bratus and Igor Pak, *Fast constructive recognition of a black box group isomorphic to S_n or A_n using Goldbach's Conjecture*, J. Symbolic Comput. **29** (2000), 33–57. MR **2001c:**11137

[8] Peter A. Brooksbank, *Constructive recognition of the finite simple classical groups*, Ph.D. thesis, University of Oregon, 2001.

[9] Peter A. Brooksbank and William M. Kantor, *On constructive recognition of a black box* PSL(d, q), Groups and Computation III (William M. Kantor and Ákos Seress, eds.), OSU Mathematical Research Institute Publications, vol. 8, Walter de Gruyter, 2001, pp. 95–111.CMP 2001:12

[10] R. D. Carmichael, *Abstract definitions of the symmetric and alternating groups and certain other permutation groups*, Quart. J. of Math. **49** (1923), 226–270.

[11] Frank Celler, Charles R. Leedham-Green, Scott H. Murray, Alice C. Niemeyer, and E. A. O'Brien, *Generating random elements of a finite group*, Comm. Algebra **23** (1995), 4931–4948. MR **96h:**20115

[12] Gene Cooperman, Larry Finkelstein, and Steven A. Linton, *Recognizing* $GL_n(2)$ *in non-standard representation*, Groups and Computation II (Larry Finkelstein and William Kantor, eds.), Amer. Math. Soc. DIMACS Series, vol. 28, (DIMACS, 1995), 1997, pp. 85–100. MR **98d:**20054

[13] H.S.M. Coxeter and W.O.J. Moser, *Generators and relations for discrete groups*, 4th ed., Ergeb. Math. Grenzgeb., vol. 14, Springer-Verlag, Berlin, Göttingen, Heidelberg, 1957. MR **81a:**20001

[14] P. Erdős and P. Turán, *On some problems of a statistical group-theory. I.*, Z. Wahrscheinlichkeitstheorie und Verw. Gebiete **4** (1965), 175–186; *II.*, Acta Math. Acad. Sci. Hungar. **18** (1967), 151–163; *III.*, Acta Math. Acad. Sci. Hungar. **18** (1967), 309–320; *IV.*, Acta Math. Acad. Sci. Hungar. **19** (1968), 413–435; *V.*, Period. Math. Hungar. **1** (1971), 5–13; *VI.*, J. Indian Math. Soc. **34** (1970), 175–192; *VII.*, Period. Math. Hungar. **2** (1972), 149–163. MR **32:**2465; MR **34:**7624; MR **35:**6743; MR **38:**1156; MR **44:**6638; MR **58:**21974; MR **56:**5470

[15] William M. Kantor and Kay Magaard, *Black box exceptional groups of Lie type*, (2002), In preparation.

[16] William M. Kantor and Ákos Seress, *Black box classical groups*, Memoirs Amer. Math. Soc. **149** (2001), no. 708. MR **2001m:**68066

[17] Ivan Niven, Herbert S. Zuckerman, and Hugh L. Montgomery, *An Introduction to the Theory of Numbers*, 5th ed., John Wiley and Sons, New York, 1991. MR **91i:**11001

[18] Richard Warlimont, *Über die Anzahl der Lösungen von* $x^n = 1$ *in der symmetrischen Gruppe* S_n, Arch. Math. **30** (1978), 591–594. MR **58:**16851

IDA Center for Communications Research, 805 Bunn Drive, Princeton, New Jersey 08540, USA

E-mail address: beals@idaccr.org

School of Mathematical Sciences, Queen Mary and Westfield College, London E1 4NS, United Kingdom

E-mail address: crlg@maths.qmw.ac.uk

Department of Mathematics and Statistics, University of Western Australia, Crawley, Western Australia 6009, Australia

E-mail address: alice@maths.uwa.edu.au

Department of Mathematics and Statistics, University of Western Australia, Crawley, Western Australia 6009, Australia

E-mail address: praeger@maths.uwa.edu.au

Department of Mathematics, The Ohio State University, Columbus, Ohio 43210, USA

E-mail address: akos@math.ohio-state.edu.