London Mathematical Society ISSN 1461–1570

A REDUCTION ALGORITHM FOR LARGE-BASE PRIMITIVE PERMUTATION GROUPS

MASKA LAW, ALICE C. NIEMEYER, CHERYL E. PRAEGER AND ÁKOS SERESS

Abstract

We present a nearly linear-time Las Vegas algorithm that, given a large-base primitive permutation group, constructs its natural imprimitive representation. A large-base primitive permutation group is a subgroup of a wreath product of symmetric groups S_n and S_r in product action on r-tuples of k-element subsets of $\{1, \ldots, n\}$, containing A_n^r. The algorithm is a randomised speed-up of a deterministic algorithm of Babai, Luks, and Seress.

1. *Introduction*

In 1971 Sims [10] introduced the fundamental data structures *base* and *strong generating set* (SGS) for computing with finite permutation groups. For a set Ω of N points, we denote the symmetric group of all permutations of Ω by $\mathrm{Sym}(\Omega)$, and also sometimes by S_N. A *base* for $G \leqslant \mathrm{Sym}(\Omega)$ is a sequence $B = (\beta_1, \ldots, \beta_m)$ of points from Ω such that the pointwise stabiliser $G_{(\beta_1,\ldots,\beta_m)}$ of $\{\beta_1, \ldots, \beta_m\}$ is the trivial group. A base B naturally defines a subgroup chain

$$G = G^{[1]} \geqslant G^{[2]} \geqslant \ldots \geqslant G^{[m]} \geqslant G^{[m+1]} = 1, \tag{1}$$

where $G^{[i]} := G_{(\beta_1,\ldots,\beta_{i-1})}$ is the pointwise stabiliser of $\{\beta_1, \ldots, \beta_{i-1}\}$. The base B is called *non-redundant* if $G^{[i+1]}$ is a proper subgroup of $G^{[i]}$ for all i with $1 \leqslant i \leqslant m$. In this case we have

$$\frac{\log |G|}{\log N} \leqslant |B| \leqslant \log |G|, \qquad \text{where } N = |\Omega|.$$

(Here, and throughout the paper, we write logarithms to base 2.)

A *strong generating set* (SGS) for G relative to B is a generating set S for G with the property that

$$\langle S \cap G^{[i]} \rangle = G^{[i]}, \qquad \text{for } 1 \leqslant i \leqslant m + 1. \tag{2}$$

Given $G = \langle X \rangle \leqslant \mathrm{Sym}(\Omega)$, Sims' algorithm constructs a non-redundant base B and an SGS S relative to B. Once S is known, it is easy to construct transversals T_i for the cosets of $G^{[i+1]}$ in $G^{[i]}$ (called transversals of $G^{[i]}$ mod $G^{[i+1]}$). These transversals can be used to compute the order of G as

$$|G| = \prod_{i=1}^{m} |T_i|,$$

and to factorise any $g \in G$ as a product $g = r_1 \cdots r_m$, with $r_i \in T_i$ for $1 \leqslant i \leqslant m$.

This project forms part of Australian Research Council Discovery Grant DP055787 that supported the first author. The fourth author was partially supported by the NSA and NSF.
Received 9 December 2005, revised 29 March 2006; *published* 16 May 2006.
2000 Mathematics Subject Classification 20B15, 20B40

https://doi.org/10.1112/S1461157000001236 Published Online by Cambridge University Press

This factorisation is unique and can be obtained by an efficient algorithm called *sifting*. Sifting can be used to test membership in G. For details, we refer to [9, Section 4.1].

Deterministic versions of Sims' algorithm run in $O(|X|N^2 \log^c |G|)$ time, and there are randomised versions that run in $O(|X|N \log^c |G|)$ time [1]. In particular, if $\log |G|$ is bounded above by a polylogarithmic function of N, then the latter running time is *nearly linear* as a function of the input length $|X|N$, namely $O((|X|N) \log^{c'}(|X|N))$. This motivated the following definition. An infinite family $\mathcal{G}$ of permutation groups is called *small-base* if every group $G \in \mathcal{G}$ of degree m satisfies $\log |G| < \log^c m$ for some fixed constant c. A family which is not small-base is called *large-base*.

For a group G in a large-base family, the term $\log^c |G|$ may dominate the timing in Sims' algorithm. In fact, no deterministic version of Sims' algorithm is known to have a worst-case running time better than $O(N^5 + |X|N^2)$ for arbitrary inputs $G = \langle X \rangle \leqslant \mathrm{Sym}(\Omega)$ with $|\Omega| = N$. The notorious N^5 barrier was broken in [2, 3] by a different kind of deterministic algorithm with running time $O(N^4 \log^c N + |X|N^2)$, that computed $|G|$ and set up a data structure to test membership in G.

The algorithm in [3] is based on principles quite different from those in Sims' original approach. It detects large alternating sections of the input group G and handles them by special methods, while the rest of G is handled by Sims' method. Recently, a randomised version of [3] was implemented in GAP [7]. This implementation is described in [8]. For this implementation, a fast detection of large alternating sections is required. Standard combinatorial reduction (action on orbits, followed by action on blocks of imprimitivity) leads to primitive groups, for which we need to perform the tasks described in Definition 1.1 below. Presenting a fast algorithm to perform these tasks is the main result of this paper. The algorithm has been implemented as part of the GAP package 'recog', described in [8].

Let $N = \binom{n}{k}^r$, for some positive integers n, r, k with $k < n/2$, and let Δ be the set of r-tuples of k-sets of $\{1, \ldots, n\}$. Then $|\Delta| = N$. The group

$$G(n, r) = S_n \wr S_r \cong S_n^r : S_r$$

acts primitively on Δ by the natural product action: for $(\delta_1, \ldots, \delta_r) \in \Delta$ and $g = (g_1, \ldots, g_r)h \in G(n, r)$, where $\delta_i \subseteq \{1, \ldots, n\}$, $g_i \in S_n$, and $h \in S_r$, we have

$$(\delta_1, \ldots, \delta_r)^g = (\delta_{1^a}^{g_{1^a}}, \ldots, \delta_{r^a}^{g_{r^a}}), \qquad \text{where } a = h^{-1}.$$

We consider the family $\mathcal{F}$ of all primitive permutation groups with the following property: $G \leqslant \mathrm{Sym}(\Omega)$ is in $\mathcal{F}$ if and only if there are positive integers n, r, k such that $|\Omega| = N = \binom{n}{k}^r$ and G is permutation isomorphic to a subgroup of $G(n, r) \leqslant \mathrm{Sym}(\Delta)$ with $A_n^r \lesssim G \lesssim S_n \wr S_r$. The significance of the family $\mathcal{F}$ is that by [5], the primitive permutation groups *not* belonging to $\mathcal{F}$ constitute a small-base family, while $\mathcal{F}$ itself is a large-base family.

DEFINITION 1.1. Let $G = \langle X \rangle$ be a primitive permutation group in $\mathcal{F}$, acting on Ω of size N, so that

$$A_n^r \lesssim G \lesssim S_n \wr S_r \qquad \text{and} \qquad N = \binom{n}{k}^r.$$

We say that G is *constructively identifiable* if there are Las Vegas algorithms for the following tasks:

(i) find nr subsets $\mathbf{J}_1, \ldots, \mathbf{J}_{nr}$ of Ω such that $\Sigma = \{\mathbf{J}_1, \ldots, \mathbf{J}_{nr}\}$ is G-invariant and G acts faithfully and transitively on Σ;

(ii) find a monomorphism $\phi : G \to \mathrm{Sym}(\Sigma)$ specified by the image of X;

and moreover, there is a deterministic algorithm for computing $g\phi$ for any given $g \in G$.

https://doi.org/10.1112/S1461157000001236 Published online by Cambridge University Press

The problem of identifying these primitive large-base groups constructively was solved by a quadratic-time deterministic algorithm in [2, 3]. Here we give a nearly linear-time, randomised solution based on ideas from these papers.

Let $\mathcal{F}_0$ be the subfamily of $\mathcal{F}$ comprising all those groups in $\mathcal{F}$ for which $n \leqslant 2rk^2$. We shall prove (see Lemma 2.1) that $\mathcal{F}_0$ is a small-base family. Then, since $\mathcal{F}$ is a large-base family, the complement $\mathcal{F} \setminus \mathcal{F}_0$ is also large-base. Thus it is sufficient to deal with $\mathcal{F} \setminus \mathcal{F}_0$. Our aim is to provide a fully analysed algorithm to identify constructively a given transitive permutation group, assuming that it belongs to $\mathcal{F} \setminus \mathcal{F}_0$. This problem is trivially solved for the natural actions of alternating and symmetric groups (that is, those groups in $\mathcal{F} \setminus \mathcal{F}_0$ with $r = k = 1$) since in this case $\Sigma = \Omega$ and ϕ can be taken as the identity map. Thus we always assume that $rk \geqslant 2$.

In this paper we prove the following theorem, where ξ is an upper bound on the time required per element to construct independent, (nearly) uniformly distributed random elements of G. As usual, since we are working in a permutation-group setting, computing an image of a point under a permutation is taken as having unit cost. The permutation domain Ω is given a total ordering, and we often sort an m-subset of Ω with respect to this ordering. This can be done at a cost of $m \log m$ comparisons of two points to determine which is greater.

THEOREM 1.2 (MAIN THEOREM). *Let $G = \langle X \rangle \in \mathcal{F} \setminus \mathcal{F}_0$ acting on Ω of size N, so that*

$$A_n^r \leqslant G \leqslant S_n \wr S_r \qquad \text{and} \qquad N = \binom{n}{k}^r,$$

for some positive integers n, k, r with $n > 2rk^2$ and $rk > 1$. Given an error probability $\varepsilon > 0$, the group G can be identified constructively, with probability at least $1 - \varepsilon$, in

$$O((\xi + |X|N + N \log N) \log N (\log N + \log \varepsilon^{-1}))$$

time. The time requirement of computing $g\phi$, for any given $g \in G$, is $O(nr^2k \log(rk)) = o(N)$, and the underlying data structures require $O(N \log N)$ memory.

The significance of this result lies in the practical application: reducing the degree of the input group from $N = \binom{n}{k}^r$ to the more easily manageable nr. Moreover (see Lemma 2.2), this action of G of degree nr is its natural imprimitive action on r blocks of size n. Note that, for $g \in G$, we can construct $g\phi$ in time $o(N)$, without reading the entire permutation representing g as an element of $\mathrm{Sym}(\Omega)$; see Step 7 in Section 3.

Note that, in Subsection 2.2, we describe an algorithm to compute the orbits of a point stabiliser in an arbitrary finite permutation group. This algorithm is of independent interest, applicable, for example, in computing blocks of imprimitivity.

2. *Some preliminaries*

Observe that, since $k < n/2$, we always have

$$\binom{n}{k} = \prod_{i=0}^{k-1} \frac{n-i}{k-i} \geqslant \left(\frac{n}{k}\right)^k > 2^k,$$

and hence $N = \binom{n}{k}^r > 2^{kr}$, whence

$$\log N > kr. \tag{3}$$

First we show that the subfamily $\mathcal{F}_0$ of $\mathcal{F}$ comprising those primitive subgroups $G \leqslant S_n \wr S_r$ in product action of degree $N = \binom{n}{k}^r$, but with $n \leqslant 2rk^2$, is small-base.

https://doi.org/10.1112/S1461157000001236 Published online by Cambridge University Press

LEMMA 2.1. *$\mathcal{F}_0$ is small-base; in fact, if $G \leqslant S_n \wr S_r$ is in $\mathcal{F}_0$, then* $\log|G| < 4\log^3 N < \log^5 N$.

Proof. Consider $G \in \mathcal{F}_0$ with $G \leqslant S_n \wr S_r$. Since $n \leqslant 2rk^2$, we have

$$|G| \leqslant (n!)^r r! < n^{nr} r^r \leqslant (2rk^2)^{2r^2k^2} r^r \leqslant (2r^2k^2)^{2r^2k^2}.$$

By (3), we have

$$\log|G| < (2\log^2 N)\log(2\log^2 N) < 2\log^2 N(2\log N) < \log^5 N$$

using $\log(2x^2) < 2x$ for $x \geqslant 1$, and $\log N > kr \geqslant 2$. □

Next we prove that, for a primitive group $G \leqslant S_n \wr S_r$ of degree $\binom{n}{k}^r$, with $n > 2rk^2$, the only transitive action of G of degree nr is its natural action.

LEMMA 2.2. *Suppose that $G \leqslant S_{nr}$ is transitive on $\Sigma = \{1, \ldots, nr\}$, where $n > 2rk^2$ and $rk > 1$, and has a normal subgroup $H \cong A_n^r$. Then H has r orbits of length n in Σ, and so G has the natural action on Σ (imprimitive if $r > 1$).*

Proof. If $r = 1$ the result is obvious, so we assume that $r \geqslant 2$. Write $H = H^{(1)} \times \ldots \times H^{(r)}$, where for each i, $H^{(i)} \cong A_n$, and let $\pi_i : H \to H^{(i)}$ denote the projection map. Since H is normal in G, the H-orbits in Σ have a constant size, say $m > 1$, so $m \mid nr$. Let $\sigma \in \Sigma$, and set

$$m = |\sigma^H| = |H : H_\sigma|.$$

Then $H_\sigma \leqslant \pi_1(H_\sigma) \times \ldots \times \pi_r(H_\sigma)$, and so $m = |H : H_\sigma|$ is divisible by $\prod_{i=1}^r |H^{(i)} : \pi_i(H_\sigma)|$. Now each proper subgroup of A_n has index at least n, and by assumption, $m \leqslant nr < n^2/2$. Thus $\pi_i(H_\sigma) = H^{(i)}$ for all but at most one i. If $\pi_i(H_\sigma) = H^{(i)}$ for all i, then $H_\sigma \cong A_n^s$ for some $s < r$ and

$$m = |H : H_\sigma| \geqslant |A_n| = \frac{1}{2}(n!),$$

contradicting the fact that $m < n^2/2$.

Hence, without loss of generality we have

$$\pi_1(H_\sigma) \lneqq H^{(1)}, \qquad \text{and} \qquad \pi_i(H_\sigma) = H^{(i)} \qquad \text{for all } i > 1.$$

If $\pi : H \to \prod_{i=2}^r H^{(i)}$ is the natural projection, then $H_\sigma \leqslant \pi_1(H_\sigma) \times \pi(H_\sigma)$, and a similar argument to that in the previous paragraph shows that $\pi(H_\sigma) = \prod_{i=2}^r H^{(i)}$, and hence that $H_\sigma = \pi_1(H_\sigma) \times \prod_{i=2}^r H^{(i)}$. Thus

$$m = |H^{(1)} : \pi_1(H_\sigma)| \geqslant n.$$

Since the normal subgroup $\prod_{i=2}^r H^{(i)}$ of H is contained in the point stabiliser H_σ, the group $\prod_{i=2}^r H^{(i)}$ fixes the H-orbit σ^H pointwise. Now H is faithful on Σ, since $H \leqslant S_{nr}$, and it follows that H is intransitive. Thus $n \leqslant m < nr$.

If $m = n$ then the result is proved, so we assume that $m > n$. Then $n < m < nr$ and $m \mid nr$. In particular $r \geqslant 3$, and so $n \geqslant 7$. Also, since $n > 2r$, we have

$$m = |H^{(1)} : \pi_1(H_\sigma)| \leqslant nr < \binom{n}{3}.$$

https://doi.org/10.1112/S1461157000001236 Published online by Cambridge University Press

By [6, Theorem 5.2A], the only proper subgroups of $H^{(1)} \cong A_n$ of index less than $\binom{n}{3}$ and greater than n are:

(i) A_{n-2} (of index $n(n-1)$);

(ii) S_{n-2} (of index $n(n-1)/2$);

(iii) $\mathrm{PSL}_3(2)$ with $(n, m) = (7, 15)$, $\mathrm{AGL}_3(2)$ with $(n, m) = (8, 15)$, or $\mathrm{P\Gamma L}_2(8)$ with $(n, m) = (9, 120)$.

All of these cases are impossible, since m must be a proper divisor of nr with $r < n/2$. □

2.1. *Schreier trees and sifting*

Schreier trees are data structures to store transversals in the point stabiliser chain defined in (1). In our algorithm, we do not construct a full point stabiliser chain for the input $G \in \mathcal{F} \setminus \mathcal{F}_0$, and we need only the first transversal for G mod G_α, for some $\alpha \in \Omega$. Therefore, we restrict our description to this special case.

For any $G \leqslant \mathrm{Sym}(\Omega)$ and $\alpha \in \Omega$, a *Schreier tree* for G mod G_α is a pair (S, T), where $S \subseteq G$ satisfies $\alpha^{\langle S \rangle} = \alpha^G$, and T is a directed labelled tree with root α. The vertex set of T is α^G. All edges of T are directed toward the root α, and the edge labels are elements of S where the directed edge from γ to δ with label s satisfies $\gamma^s = \delta$. If γ is a vertex of T, then the sequence $(s_1, \ldots, s_m)$ of the edge labels along the unique path from γ to α in T determines a word in the elements of S such that the product $r_\gamma = s_1 \cdots s_m$ of these permutations maps γ to α. Thus a Schreier tree (S, T) defines the *inverses* of a set of right coset representatives for G_α in G.

For $g \in G$ with $\alpha^g = \gamma$, computing r_γ and $g_\alpha := g r_\gamma \in G_\alpha$ is called the *sifting* of g into G_α.

The time requirement for sifting $g \in G$ with $\alpha^g = \gamma$ is proportional to the length of the path from γ to α in T. Therefore, given S, we want T to have as small a depth as possible. To this end, we may construct T as a *breadth-first-search* tree: level $L_0 := \{\alpha\}$, and if level L_i is already constructed, L_{i+1} is the set of those $\gamma \in \alpha^G \setminus \bigcup_{j=0}^{i} L_j$ that are of the form $\gamma = \delta^s$ for some $\delta \in L_i$ and $s \in S^{-1}$.

2.2. *Constructing the orbits of a point stabiliser*

In our algorithm, we require particular orbits of a point stabiliser in a large-base primitive permutation group. In this subsection we present a Las Vegas algorithm that constructs all such orbits in nearly linear time. The results are independent of our special setting, holding true for any finite permutation group.

Thus throughout this subsection, $G \leqslant \mathrm{Sym}(\Omega)$ is an arbitrary permutation group on a finite set Ω of N points, and $\alpha \in \Omega$. We use [9, Theorem 4.4.6], which is a strengthened version of the main result of [4]. First we construct a subgroup of G_α which, with high probability, has the same orbits as G_α.

LEMMA 2.3. *Let ε satisfy $0 < \varepsilon < 1$. For G as above, $O(\log N(\log N + \log(\varepsilon^{-1})))$ independent, uniformly distributed random elements of G_α, generate a subgroup $H \leqslant G_\alpha$ which has the same orbits as G_α in Ω with probability at least $1 - \varepsilon$.*

Proof. Let Δ_j be an orbit of G_α of length $m_j > 1$. By [9, Theorem 4.4.6] the probability that any subgroup of G_α generated by at least $8 \log m_j + 16$ independent, uniformly distributed random elements of G_α has Δ_j as an orbit is at least $1 - m_j^{-0.29}$. Since $m_j \leqslant N$, we

https://doi.org/10.1112/S1461157000001236 Published online by Cambridge University Press

can choose $c > 1$ such that $c(8 \log N + 16)$ independent, uniformly distributed, random elements of G_α generate a subgroup that has the same orbits as G_α with accumulated probability of failure at most

$$\sum_{m_i} m_i^{-0.29c},$$

where the sum is over all G_α-orbits in Ω of lengths greater than 1. Since the worst case for the error occurs in the situation where $m_i = 2$ for all orbits apart from $\{\alpha\}$, the accumulated probability of failure is less than

$$\frac{N}{2} \cdot 2^{-0.29c}.$$

For any $\varepsilon > 0$, to ensure that this probability of failure is at most ε, we must take

$$c \geqslant \frac{\log N + \log(\varepsilon^{-1}) - 1}{0.29}.$$ □

To apply the previous lemma, we need to be able to construct independent, uniformly distributed, random elements of G_α. We do this by constructing a (shallow) Schreier tree for G mod G_α (see Subsection 2.1). We can then obtain random elements of G_α by sifting random elements of G through this Schreier tree; this, if the tree has depth d, involves at most $d + 1$ permutation multiplications for each element.

LEMMA 2.4. *For G as above, and ε satisfying $0 < \varepsilon < 1$, there is a Las Vegas algorithm that, with probability at least $1 - \varepsilon$, constructs the orbits of G_α in*

$$O((\xi + N \log N)(\log N + \log(\varepsilon^{-1})) \log N)$$

time, where ξ is as defined before Theorem 1.2.

Proof. By [9, Theorem 4.4.6], for $c_1 \geqslant 1$, $c_1(8 \log N + 16)$ random elements of G construct a breadth-first-search Schreier tree for G mod G_α, of depth at most $2 \log N + 4$, with probability of failure less than $N^{-0.29c_1}$. For any $\varepsilon > 0$, to ensure that this probability of failure is at most $\varepsilon/2$, we must take

$$c_1 \geqslant \frac{1 + \log(\varepsilon^{-1})}{0.29 \log N},$$

and since c_1 must be at least 1, we take

$$\begin{aligned} c_1 &= \max\left\{\frac{1 + \log(\varepsilon^{-1})}{0.29 \log N}, 1\right\} \\ &\leqslant \max\left\{\frac{1 + \log(\varepsilon^{-1})}{0.29}, 1\right\} = O(\log(\varepsilon^{-1})). \end{aligned}$$

Thus we require a set X of $O(\log N \log(\varepsilon^{-1}))$ random elements of G, each of which is then applied to the N points of Ω, to construct a tree reaching every coset of G mod G_α. Replacing the elements of X by their inverses, but keeping the labelling of the tree edges, we obtain the Schreier tree for G mod G_α. The cost of constructing the Schreier tree is therefore $O((\xi + N) \log N \log(\varepsilon^{-1}))$. This tree can be stored in an array of length N (see [9, Section 4.1]).

Assuming that the Schreier tree has been successfully constructed, we now attempt to construct a subgroup H of G_α with the same orbits as G_α in Ω, as in Lemma 2.3. Sifting

https://doi.org/10.1112/S1461157000001236 Published online by Cambridge University Press

$c_2(8 \log N + 16)$ random elements of G, with

$$c_2 = \frac{\log N + \log(\varepsilon^{-1})}{0.29} = O(\log N + \log(\varepsilon^{-1})),$$

through the Schreier tree, we obtain independent, uniformly distributed, random elements of G_α, that generate a subgroup H of G_α with the same orbits as G_α, with a probability of failure less than $\varepsilon/2$. The cost of selecting and sifting the $O((\log N + \log(\varepsilon^{-1})) \log N)$ random elements through the Schreier tree of depth $O(\log N)$ is therefore

$$O((\xi + N \log N)(\log N + \log(\varepsilon^{-1})) \log N).$$

Taking these generators, we may then construct the orbits of G_α in

$$O(N \log N(\log N + \log(\varepsilon^{-1})))$$

time (see [9, Section 2.1.1]). The overall probability of failure is less than $\varepsilon/2+\varepsilon/2 = \varepsilon$. □

2.3. *More details on the action of large-base primitive groups*

As noted in Section 1, the family of large-base primitive groups consists of certain subgroups of wreath products in product action, namely subgroups G of

$$G(n, r) = S_n \wr S_r \cong S_n^r : S_r < \mathrm{Sym}(\Omega),$$

where we identify Ω with the set of r-tuples of k-element subsets of $\{1, \ldots, n\}$. In particular, G must contain the socle A_n^r of $G(n, r)$ and G must act transitively on the n copies of A_n in its socle by conjugation. Thus $N = |\Omega| = \binom{n}{k}^r$ and, as discussed in Section 1, we may assume that $n > 2rk^2$, $rk > 1$, and $k < n/2$.

We therefore may view a point η of Ω as an r-tuple $\eta = (\eta_1, \ldots, \eta_r)$ where $\eta_i \subseteq \{1, \ldots, n\}$ and $|\eta_i| = k$ for $i \in \{1, \ldots, r\}$. For $\eta \in \Omega$, let $\eta[s]$ denote the k-set in the sth component of η. We may then describe η by the set P_η of (*letter, component*)*-pairs* (m, s), where $m \in \{1, \ldots, n\}$ and $s \in \{1, \ldots, r\}$, such that $m \in \eta[s]$. Thus

$$(m, s) \in P_\eta \qquad \text{if and only if} \qquad m \in \eta[s]. \tag{4}$$

For $\eta, \zeta \in \Omega$, we say that η *meets* ζ if $\eta[s] \cap \zeta[s] \neq \emptyset$ for some $s \in \{1, \ldots, r\}$ or, equivalently, if $P_\eta \cap P_\zeta \neq \emptyset$.

Since G is primitive on Ω, its image under the natural homomorphism from $G(n, r)$ to S_r is a transitive subgroup of S_r. Thus, with reference to the product action of $G(n, r)$ defined in Section 1, G acts transitively on the components of the points of Ω. Choose a point $\alpha \in \Omega$. Our algorithm makes essential use of two special G_α-orbits, defined as follows.

Definition 2.5. For $\alpha \in \Omega$, define $\Gamma(\alpha)$ to be the subset of all points in Ω which differ from α in only one component, and in that component differ by only one letter; that is,

$$\Gamma(\alpha) = \{\beta \in \Omega : \exists c \text{ such that } \forall t \in \{1, \ldots, r\} \setminus \{c\},\ \beta[t] = \alpha[t],\ |\beta[c] \cap \alpha[c]| = k - 1\}.$$

Define $\Delta(\alpha)$ to be the subset of all points in Ω which do not *meet* α; that is,

$$\Delta(\alpha) = \{\beta \in \Omega : \forall\, t \in \{1, \ldots, r\},\ \beta[t] \cap \alpha[t] = \emptyset\}.$$

Lemma 2.6 (*see* [3, Lemma 2.10]). *For G as above, $\Gamma(\alpha)$ is the shortest, and $\Delta(\alpha)$ is the longest G_α-orbit in $\Omega \setminus \{\alpha\}$.*

Next we define a family of nr subsets of Ω which become our new point set for the imprimitive action of G.

https://doi.org/10.1112/S1461157000001236 Published online by Cambridge University Press

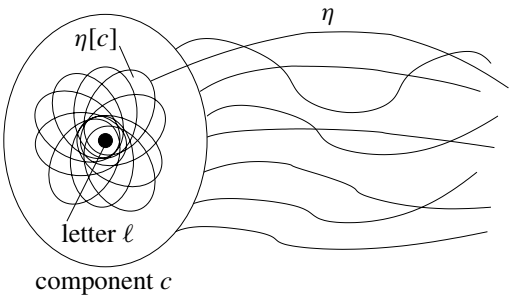

Figure 1: A representation of a point η of the *jellyfish* $\mathbf{J}(\ell, c)$

DEFINITION 2.7. For each $c \in \{1, \ldots, r\}$ and $\ell \in \{1, \ldots, n\}$, define the *jellyfish* $\mathbf{J}(\ell, c)$ to be the subset of all points of Ω whose cth component contains the letter ℓ; that is,

$$\mathbf{J}(\ell, c) = \{\eta \in \Omega : (\ell, c) \in P_\eta\}.$$

Since $k < n/2$, there exist points $\eta, \zeta \in \mathbf{J}(\ell, c)$ such that $P_\eta \cap P_\zeta = \{(\ell, c)\}$, and hence

$$\bigcap_{\eta \in \mathbf{J}(\ell,c)} P_\eta = \{(\ell, c)\}.$$

Thus, a jellyfish determines a unique (letter, component)-pair. Note that each point of Ω lies in exactly rk jellyfish, and the cardinality of each jellyfish is

$$|\mathbf{J}(\ell, c)| = \binom{n-1}{k-1}\binom{n}{k}^{r-1} = N\frac{k}{n}.$$

Speaking informally, the set $\mathbf{J}(\ell, c)$ focuses attention on 'what is happening' at a particular component and around a particular letter. We found a representation of $\mathbf{J}(\ell, c)$, as in Figure 1, helpful in identifying important properties of the set, and this in turn suggested the name *jellyfish*. Since the sets $\mathbf{J}(\ell, c)$ were central to our algorithm, a name for them was required, and the name *jellyfish* 'stuck'. Perhaps Figure 1 may also be helpful to the reader.

The group $G(n, r)$ permutes the set of jellyfish in a natural way: for $g = (g_1, \ldots, g_r)h \in G(n, r)$, where each $g_i \in S_n$ and $h \in S_r$, we have $\mathbf{J}(\ell, c)^g = \mathbf{J}(\ell^{g_c}, c^h)$. Since G projects to a transitive subgroup of S_r, and since G contains A_n^r, it follows that G is transitive on the set Σ of all the nr jellyfish. Moreover, we have the following lemma.

LEMMA 2.8. *The group G acts faithfully on the set Σ of jellyfish; moreover, this action is equivalent to the natural imprimitive action of G with r blocks of size n.*

Proof. Let K be a non-trivial normal subgroup of G. Since G is primitive on Ω, K is transitive on Ω, and so K fixes no jellyfish setwise. Thus K acts non-trivially on Σ. Taking $K = A_n^r$, by Lemma 2.2, the induced action of G on the set Σ of all jellyfish is the natural action with r blocks of size n (imprimitive if $r > 1$). □

Our approach is to construct one jellyfish, from which we can construct all jellyfish as the orbit under G of this first jellyfish. The main ingredient from a computational point of view is the use of G_α-orbits. In particular, we make use of the shortest and longest G_α-orbits in $\Omega \setminus \{\alpha\}$ given in Lemma 2.6. For $\beta \in \Gamma(\alpha)$ such that $\beta[c] \setminus \alpha[c] = \{\ell\}$, we show in Lemma 4.1 that $\mathbf{J}(\ell, c)$ may be obtained by removing from Ω the subsets $\Delta(\gamma)$ for certain points $\gamma \in \Delta(\alpha) \setminus \Delta(\beta)$.

https://doi.org/10.1112/S1461157000001236 Published online by Cambridge University Press

3. *Overview of our approach*

In this section we outline our solution strategy, describing the way in which the geometric structure of the assumed permutation domain, described in Subsection 2.3, can be exploited computationally to obtain a set of 'new points' upon which the given large-base primitive permutation group acts faithfully and transitively.

We outline the steps of the algorithm described and analysed in this paper. Given a permutation group $G = \langle X \rangle$ acting primitively on $\Omega = \{1, \ldots, N\}$, where

$$N = \binom{n}{k}^r \qquad \text{and} \qquad A_n^r \leqslant G \leqslant G(n, r) = S_n \wr S_r,$$

we construct a homomorphism (by images of generators) from G into S_{nr}. Note that, although we assume that G is contained in $G(n, r)$ in its product action, we do not have available an identification of the points of Ω with r-tuples of k-sets from $\{1, \ldots, n\}$.

1. Choose a point $\alpha \in \Omega$, and construct a shallow Schreier tree for G acting on Ω mod G_α, as in Subsection 2.2.

2. By sifting random elements of G through the Schreier tree to obtain random elements of G_α (see Subsections 2.1 and 2.2), construct a subgroup $H \leqslant G_\alpha$ which has the same orbit structure as G_α, and so construct the shortest G_α-orbit $\Gamma(\alpha)$, and the longest G_α-orbit $\Delta(\alpha)$ in $\Omega \setminus \{\alpha\}$.

3. Choose a point $\beta \in \Gamma(\alpha)$ (where, for some $c \in \{1, \ldots, n\}$ and $\ell \in \{1, \ldots, r\}$, $\beta[c] \setminus \alpha[c] = \{\ell\}$). By taking random elements of $\Delta(\alpha) \setminus \Delta(\beta)$, obtain a subset $\mathbf{I}_1$ such that $\Omega \setminus \bigcup_{\gamma \in \mathbf{I}_1} \Delta(\gamma)$ is a jellyfish, namely $\mathbf{J}_1 = \mathbf{J}(\ell, c)$.

4. Construct the set Σ of all nr jellyfish as the G-orbit of the first jellyfish $\mathbf{J}_1$, and check that Σ is G-invariant. Use the images of $\mathbf{I}_1$ corresponding to the jellyfish already constructed to help determine whether a new jellyfish image is equal to one already constructed.

5. During the orbit calculation, record — for each point $\gamma \in \Omega$ — a list of the indices j of the rk jellyfish $\mathbf{J}_j$ containing γ (as rows of Table $T1$). Also record, for each jellyfish $\mathbf{J}_j$, the points of $\mathbf{J}_j$ in increasing order (as rows of Table $T2$), the points of the set $\mathbf{I}_j$ corresponding to $\mathbf{J}_j$ (as rows of Table $T3$), and information about how $\mathbf{J}_j$ was obtained as an image of some previously constructed jellyfish, namely an integer i and generator $g \in X$ such that $\mathbf{J}_j = \mathbf{J}_i^g$ (as rows of Table $T4$).

6. For each jellyfish $\mathbf{J}_i$, reduce the set $\mathbf{I}_i$ to a pair $\mathbf{I}_i = \{\gamma_i, \delta_i\}$ of points of Ω such that $P_{\gamma_i} \cap P_{\delta_i} = \{(m, s)\}$, where $\mathbf{J}_i = \mathbf{J}(m, s)$.

7. Rewrite each of the generators in X as permutations of degree nr by determining the images under each generator of each of the nr jellyfish. The image under $g \in X$ of $\mathbf{J}_i$ is calculated by taking the images under g of the two points in $\mathbf{I}_i$ and intersecting the corresponding rows of Table $T1$.
   This same method is used, for an arbitrary element $g \in G$, to compute the permutation of $\{1, \ldots, nr\}$ corresponding to g. Note that this requires $O(nr) = o(N)$ image computations under the permutation g. In particular we do not need even to read the whole of the permutation of g in its action on the N points of Ω into memory.

https://doi.org/10.1112/S1461157000001236 Published online by Cambridge University Press

4. *Constructing the first jellyfish*

In this section we are concerned with Step 3 of the outline in Section 3. Suppose that we have already constructed the orbits of G_α. Pick any $\beta \in \Gamma(\alpha)$ (see Definition 2.5 and Lemma 2.6); then there exist a component c and a letter ℓ such that α and β differ only in component c, and $\beta[c] \setminus \alpha[c] = \{\ell\}$. We want to construct the jellyfish $\mathbf{J}(\ell, c)$: the set of all points in Ω whose cth component contains the letter ℓ. Note that for any $\gamma \in \Omega$, we have $\gamma \in \Delta(\alpha) \setminus \Delta(\beta)$ if and only if $\gamma[d]$ is disjoint from $\alpha[d] = \beta[d]$ in component d for all $d \neq c$, $\ell \in \gamma[c]$, and $\gamma[c]$ is disjoint from $\alpha[c]$. Thus, for $\gamma \in \Delta(\alpha) \setminus \Delta(\beta)$ we have $\gamma \in \mathbf{J}(\ell, c)$ and $\Delta(\gamma) \cap \mathbf{J}(\ell, c) = \emptyset$. We construct $\mathbf{J}(\ell, c)$ by removing $\Delta(\gamma)$ from Ω for various points $\gamma \in \Delta(\alpha) \setminus \Delta(\beta)$.

LEMMA 4.1. *Let α, β, ℓ, c be such that $\beta \in \Gamma(\alpha)$ and $\alpha[c] \setminus \beta[c] = \{\ell\}$, let ε satisfy $0 < \varepsilon < 1$, and let $\mathbf{I}$ be a set of $\lceil \log(N/\varepsilon) \rceil$ uniformly distributed, randomly chosen $\gamma \in \Delta(\alpha) \setminus \Delta(\beta)$. Then the probability that*

$$\mathbf{J}(\ell, c) = \Omega \setminus \bigcup_{\gamma \in \mathbf{I}} \Delta(\gamma)$$

is at least $1 - \varepsilon$.

Proof. Fix a point $\delta \in \Omega$ such that $\delta \notin \mathbf{J}(\ell, c)$, so $\ell \notin \delta[c]$. For a uniformly distributed randomly chosen $\gamma \in \Delta(\alpha) \setminus \Delta(\beta)$, for $d \neq c$ we have $\gamma[d] \cap \alpha[d] = \emptyset$, and so the probability that $\gamma[d]$ is disjoint from $\alpha[d] \cup \delta[d]$ is at least

$$\frac{\binom{n-2k}{k}}{\binom{n-k}{k}}.$$

Also, $\gamma[c] \cap \alpha[c] = \emptyset$ and $\ell \in \gamma[c]$, and so the probability that $\gamma[c] \setminus \{\ell\}$ is disjoint from $\alpha[c] \cup \delta[c] \cup \{\ell\}$ is at least

$$\frac{\binom{n-2k-1}{k-1}}{\binom{n-k-1}{k-1}}.$$

Now $\delta \in \Delta(\gamma)$ if and only if $\delta[d] \cap \gamma[d] = \emptyset$ for all d, and hence the probability of this is at least

$$\begin{aligned}
\frac{\binom{n-2k-1}{k-1}\binom{n-2k}{k}^{r-1}}{\binom{n-k-1}{k-1}\binom{n-k}{k}^{r-1}} &= \left(\prod_{i=0}^{k-2} \frac{n-2k-1-i}{n-k-1-i}\right)\left(\prod_{i=0}^{k-1} \frac{n-2k-i}{n-k-i}\right)^{r-1} \\
&\geqslant \left(\frac{n-3k}{n-2k}\right)^{k-1+k(r-1)} \\
&\geqslant \left(\frac{2rk^2-3k}{2rk^2-2k}\right)^{rk-1}
\end{aligned}$$

using the inequality $n > 2rk^2$. This is equal to

$$\left(\left(1 - \frac{1}{2rk-2}\right)^{2rk-2}\right)^{1/2} \geqslant \frac{1}{2}$$

since $(1 - 1/x)^x \geqslant 1/4$ for $x \geqslant 2$. Hence the probability that, for a set $\mathbf{I}$ of $\lceil \log(N/\varepsilon) \rceil$ uniformly distributed randomly chosen $\gamma \in \Delta(\alpha) \setminus \Delta(\beta)$, the point $\delta \notin \bigcup_{\gamma \in \mathbf{I}} \Delta(\gamma)$ is at most

$$\left(\tfrac{1}{2}\right)^{\lceil (\log N/\varepsilon) \rceil} \leqslant \varepsilon/N.$$

Since this holds for all $\delta \notin \mathbf{J}(\ell, c)$, the probability that $\mathbf{J}(\ell, c) = \Omega \setminus \bigcup_{\gamma \in \mathbf{I}} \Delta(\gamma)$ is at least $1 - \varepsilon$. □

https://doi.org/10.1112/S1461157000001236 Published online by Cambridge University Press

LEMMA 4.2. *For ε satisfying $0 < \varepsilon < 1$, given the orbits of G_α and the Schreier tree (S, T) for G mod G_α used to construct these orbits in Lemma 2.4, we can construct a jellyfish, with probability at least $1 - \varepsilon$, in $O(N \log N(\log N + \log(\varepsilon^{-1}))$ time.*

Proof. Fix $\beta \in \Gamma(\alpha)$, and let c and ℓ be such that $\beta[c] \setminus \alpha[c] = \{\ell\}$. The path from β to α in the Schreier tree T can be constructed in $O(\log N)$ time, and the product g^{-1} of the edge labels along this path can be computed in $O(N \log N)$ time. In a further $O(N)$ time we can compute g and $\Delta(\beta) = \Delta(\alpha)^g$. Using the characteristic functions of $\Delta(\alpha)$ and $\Delta(\beta)$ we can construct the set difference $\Delta(\alpha) \setminus \Delta(\beta)$ in $O(N)$ time. Now let $\mathbf{I}$ be a set of $\lceil \log(N/\varepsilon) \rceil$ random selections from $\Delta(\alpha) \setminus \Delta(\beta)$. By Lemma 4.1, we can construct the jellyfish $\mathbf{J}(\ell, c)$ as $\Omega \setminus \bigcup_{\gamma \in \mathbf{I}} \Delta(\gamma)$, with probability of success at least $1 - \varepsilon$. Using the same approach as for $\Delta(\beta)$ above, the construction of each $\Delta(\gamma)$ and subsequent removal from Ω takes $O(N \log N)$ time. Hence we require $O(N \log N(\log N + \log(\varepsilon^{-1})))$ time overall. □

5. *Constructing all jellyfish*

Let $\mathbf{I}_1$ be the set of random points of $\Delta(\alpha) \setminus \Delta(\beta)$ used in the proof of Lemma 4.2 to construct the first jellyfish $\mathbf{J}_1$, so that

$$\mathbf{J}_1 = \Omega \setminus \bigcup_{\gamma \in \mathbf{I}_1} \Delta(\gamma).$$

Recall that $\mathbf{I}_1 \subseteq \Delta(\alpha) \setminus \Delta(\beta) \subseteq \mathbf{J}_1$. For a jellyfish $\mathbf{J}$, a subset $\mathbf{I} \subseteq \mathbf{J}$ is called an *identifying set* for $\mathbf{J}$ if $\mathbf{J} = \Omega \setminus \bigcup_{\gamma \in \mathbf{I}} \Delta(\gamma)$.

Given the jellyfish $\mathbf{J}_1$ and its identifying set $\mathbf{I}_1$, we can construct all jellyfish by computing the orbit of $\mathbf{J}_1$ under G (since by assumption, G acts transitively on the set of all jellyfish). In order to determine whether the image of a jellyfish under a particular generator of G is a jellyfish that we have previously encountered, we use the following lemma.

LEMMA 5.1. *For a jellyfish $\mathbf{J}$ with identifying set $\mathbf{I}$, there is a unique (letter, component)-pair common to all $\gamma \in \mathbf{I}$; that is, $|\bigcap_{\gamma \in \mathbf{I}} P_\gamma| = 1$.*

Proof. Suppose that $\mathbf{J} = \mathbf{J}(\ell, c)$, so that $\ell \in \gamma[c]$ for all $\gamma \in \mathbf{I} \subseteq \mathbf{J}$. Then $(\ell, c) \in \bigcap_{\gamma \in \mathbf{I}} P_\gamma$. If $(m, d) \in \bigcap_{\gamma \in \mathbf{I}} P_\gamma$ with $(m, d) \neq (\ell, c)$, then $m \in \gamma[d]$ for all $\gamma \in \mathbf{I}$, and so $\{\delta \in \Omega : m \in \delta[d]\}$ is disjoint from $\Delta(\gamma)$ for all $\gamma \in \mathbf{I}$; that is,

$$\{\delta \in \Omega : m \in \delta[d]\} \subset \Omega \setminus \bigcup_{\gamma \in \mathbf{I}} \Delta(\gamma).$$

However, since $(m, d) \neq (\ell, c)$ there exist points $\delta \in \Omega$ such that $m \in \delta[d]$ and $\ell \notin \delta[c]$. Such points lie in $\Omega \setminus \bigcup_{\gamma \in \mathbf{I}} \Delta(\gamma)$ (since $m \in \delta[d]$), but do not lie in $\mathbf{J}(\ell, c)$ (since $\ell \notin \delta[c]$). Thus $\Omega \setminus \bigcup_{\gamma \in \mathbf{I}} \Delta(\gamma) \neq \mathbf{J}(\ell, c)$, which is a contradiction. □

This lemma implies that an identifying set $\mathbf{I}$ of a jellyfish cannot be a subset of a second distinct jellyfish, since in that case we would have $|\bigcap_{\gamma \in \mathbf{I}} P_\gamma| \geqslant 2$. So, given the first jellyfish $\mathbf{J}_1$ and its identifying set $\mathbf{I}_1$, we define the identifying set of a newly constructed jellyfish $\mathbf{J}_1^g$ for $g \in G$ to be $\mathbf{I}_1^g$, and we use such identifying sets to determine efficiently whether a jellyfish image $\mathbf{J}_1^g$ is equal to any jellyfish already computed.

LEMMA 5.2. *Given the jellyfish $\mathbf{J}_1$ and its identifying set $\mathbf{I}_1$, we can construct all jellyfish, with an identifying set for each, and can prove that the set of jellyfish is G-invariant, in $O(|X|N \log N(\log N + |\mathbf{I}_1|))$ time.*

https://doi.org/10.1112/S1461157000001236 Published online by Cambridge University Press

In the following proof we work with the jellyfish and identifying sets as lists of elements of Ω, of length $|\mathbf{J}|$ and $|\mathbf{I}|$ respectively. Each of the jellyfish constructed is returned as a list of its elements in increasing order.

Proof. We construct the orbit of the given jellyfish $\mathbf{J}_1$, and apply to each member $\mathbf{J}$ of the orbit the following test, for each $g \in X$. First, we construct the image $\mathbf{J}^g$ in $O(|\mathbf{J}|)$ time. Next, we sort the image $\mathbf{J}^g$, which takes $O(|\mathbf{J}| \log |\mathbf{J}|)$ comparisons. For each of the (less than nr) identifying sets $\mathbf{I}$ of the previously constructed jellyfish, we use at most $|\mathbf{I}|$ binary searches in $\mathbf{J}^g$ to check whether $\mathbf{I}$ is a subset of $\mathbf{J}^g$. Since all the $\mathbf{I}$ are images of $\mathbf{I}_1$, this requires $O(nr|\mathbf{I}_1| \log |\mathbf{J}|)$ time. If an identifying set $\mathbf{I}'$ for a jellyfish $\mathbf{J}'$ is found to lie within $\mathbf{J}^g$, we make certain that $\mathbf{J}^g$ is not new by comparing the two entire jellyfish $\mathbf{J}^g$ and $\mathbf{J}'$. (This test, carried out for all $g \in X$, guarantees that G leaves the set of jellyfish invariant.) By Lemma 5.1 this occurs for at most one of the $\mathbf{I}'$, and since all the jellyfish are stored as sorted lists, this requires $O(|\mathbf{J}|)$ comparisons. For each new jellyfish $\mathbf{J}^g$ constructed, we also compute an identifying set for it as the image under g of the identifying set for $\mathbf{J}$ in $O(|\mathbf{I}|)$ time. After applying this procedure to each of the nr jellyfish $\mathbf{J}$ in the orbit, and for each generator $g \in X$, we will have completed the required computation, and will have checked that the set of jellyfish is G-invariant. The total cost of the procedure is at most

$$O\big(nr|X|\big(|\mathbf{J}| \log |\mathbf{J}| + nr|\mathbf{I}_1| \log |\mathbf{J}|\big)\big).$$

Since $|\mathbf{J}| = N(k/n)$, this is $O(|X| \log N(Nrk + n^2r^2|\mathbf{I}_1|))$. By (3), $\log N > rk$. Also, since $rk \geqslant 2$, we have $N = \binom{n}{k}^r \geqslant \frac{1}{4}n^2r^2$, the worst case being when $k = 1, r = 2$. Thus the cost is at most $O(|X|N \log N(\log N + |\mathbf{I}_1|))$. □

6. *Constructing the homomorphism*

In this section we describe how to construct the homomorphism ϕ from G into S_{nr}, by rewriting the generators of G as permutations of the set Σ of nr jellyfish.

6.1. *Jellyfish numbering — the tables*

During the construction of the orbit of all jellyfish, we maintain tables $T1$, $T2$, $T3$ and $T4$. After completing the computation, Table $T1$ has N rows, with row i being a list in increasing order of those indices j such that the ith point lies in the jth jellyfish $\mathbf{J}_j$. Table $T2$ is a list of the jellyfish, the jth row consisting of the points of $\mathbf{J}_j$ in increasing order, while Table $T3$ is a list of identifying sets for the jellyfish, the jth row consisting of the points of an identifying set for $\mathbf{J}_j$ in increasing order. Table $T4$ is simply a list of how each jellyfish was constructed from a previously constructed jellyfish, and contains in row j the number $i < j$ and the generator $g \in X$ such that $\mathbf{J}_j = \mathbf{J}_i^g$. Tables $T3$ and $T4$ can be considered as extensions of Table $T2$.

The tables are constructed as follows. When a new jellyfish, say $\mathbf{J}_j$, is constructed, say as $\mathbf{J}_i^g$, the sorted list of points in $\mathbf{J}_j$ becomes row j of $T2$, and its identifying set is the corresponding row of $T3$. The pair (i, g) is stored in row j of $T4$. For each new jellyfish constructed in the orbit computation, the number of this new jellyfish is added to the corresponding rows of $T1$ representing each of the points in this jellyfish. Each row of $T1$ then contains the numbers of the jellyfish which contain that point. The jellyfish numbers, from 1 to nr, are determined simply by the order in which the jellyfish are constructed, and the corresponding identifying sets then have the same numbering.

https://doi.org/10.1112/S1461157000001236 Published online by Cambridge University Press

6.2. *Identifying pairs*

After having constructed all jellyfish and the Tables $T1$, $T2$, $T3$ and $T4$, we construct a pair of points which uniquely identifies the first jellyfish $\mathbf{J}_1$. This is a pair $\{\gamma, \delta\}$ of points of $\mathbf{J}_1$ with the property that γ and δ meet in only one component, and meet only once in that component; that is, $|P_\gamma \cap P_\delta| = 1$. The intersection $P_\gamma \cap P_\delta$ is the unique (letter, component)-pair represented by this jellyfish; that is to say, $\mathbf{J}_1 = \mathbf{J}(\ell, c)$ where $P_\gamma \cap P_\delta = \{(\ell, c)\}$. Since $k < n/2$, such a pair $\{\gamma, \delta\}$ exists, and we can check this property by checking that 1 is the only index in common in row γ and row δ of Table $T1$.

Given an identifying pair for the first jellyfish, we can then use Table $T4$ to construct identifying pairs for each of the other jellyfish, since the image of an identifying pair is again an identifying pair for the jellyfish image. These identifying pairs can then replace the identifying sets for each of the jellyfish, and are used as a more efficient way of rewriting group elements in the next subsection.

To find an identifying pair for the first jellyfish $\mathbf{J}_1$, we choose a point $\gamma \in \mathbf{J}_1$, and randomly search for a second point δ. As we remarked above, such points δ exist for any given γ, and we now estimate the probability of finding this second point.

LEMMA 6.1. *For ε satisfying $0 < \varepsilon < 1$, a pair of points which uniquely identifies a given jellyfish can be found in $O(rk \log(\varepsilon^{-1}))$ time, with probability at least $1 - \varepsilon$. Given an identifying pair for the first jellyfish, identifying pairs can be constructed for all remaining jellyfish in additional $O(nr)$ time. The total time required is less than $O(N + \log N \log(\varepsilon^{-1}))$.*

Proof. Given a point $\gamma \in \mathbf{J}(\ell, c)$, let p be the probability that a random point δ of $\mathbf{J}(\ell, c)\backslash\{\gamma\}$ forms an identifying pair $\{\gamma, \delta\}$ for this jellyfish (that is, that $P_\gamma \cap P_\delta = \{(\ell, c)\}$). Then

$$\begin{aligned} p &= \frac{\binom{n-k}{k-1}\binom{n-k}{k}^{r-1}}{\binom{n-1}{k-1}\binom{n}{k}^{r-1}} = \left(\prod_{i=0}^{k-2} \frac{n-k-i}{n-1-i}\right)\left(\prod_{i=0}^{k-1} \frac{n-k-i}{n-i}\right)^{r-1} \\ &\geqslant \left(\frac{n-2k}{n-k}\right)^{k-1+k(r-1)} > \left(\frac{2rk^2-2k}{2rk^2-k}\right)^{rk-1} \\ &= \left(\left(1 - \frac{1}{2rk-1}\right)^{2rk-1}\right)^{(rk-1)/(2rk-1)} \\ &> \left(\left(1-\frac{1}{3}\right)^3\right)^{1/2} > \frac{1}{2} \end{aligned}$$

since $2rk - 1 \geqslant 3$ and $(rk-1)/(2rk-1) < 1/2$.

Making m such selections therefore gives an accumulated probability of failure of at most $(1/2)^m$. For a given $\varepsilon > 0$, to ensure that this probability of failure is at most ε, we must make at least $m = \lceil \log(\varepsilon^{-1}) \rceil$ random selections. To check that a pair of points forms an identifying pair, we simply intersect the corresponding rows of Table $T1$, which requires rk comparisons (as the rows of Table $T1$ are already sorted). Thus an identifying pair for a given jellyfish can be found in time $O(rk \log(\varepsilon^{-1}))$. Since $\log N > rk$ and $N > nr$, the total time required is less than $O(N + \log N \log(\varepsilon^{-1}))$. □

6.3. *Rewriting group elements*

We are now in a position to construct the homomorphism ϕ from G into S_{nr}. We do this by computing, for each generator $g \in X$, the image of each jellyfish under g, thus writing

https://doi.org/10.1112/S1461157000001236 Published online by Cambridge University Press

g as a permutation of the set Σ of nr jellyfish. This procedure requires only Table $T1$ and the list of identifying pairs.

LEMMA 6.2. *An element $g \in G$ can be written as a permutation of the set Σ of nr jellyfish in $O(nr^2k\log(rk)) = o(N)$ time.*

Proof. For an element $g \in G$, we compute the image of each jellyfish under g as follows. Let $\mathbf{J}_i$ be a jellyfish with identifying pair $\{\gamma_i, \delta_i\}$. We first compute the image of γ_i and δ_i under g to obtain two points γ_i^g, δ_i^g (at a cost of two image computations). The rows of Table $T1$ corresponding to γ_i^g and δ_i^g intersect in a unique jellyfish number j, since $\{\gamma_i, \delta_i\}$ is an identifying pair for $\mathbf{J}_i$. The number j can be computed at a cost of $O(rk\log(rk))$ comparisons, and j is then the image of i under g. In this way, each generator can be written as a permutation of $\{1, \ldots, nr\}$ at a cost of $O(nr^2k\log(rk))$. By (3), $rk < \log N$, and as $rk \geqslant 2$ we have $\binom{n}{2} \leqslant N$. Hence

$$O(nr^2k\log(rk)) = O(N^{1/2}\log^2 N\log\log N) = o(N). \qquad \square$$

7. *Proof of Theorem 1.2*

Let $G = \langle X\rangle$, and let ε be as in Theorem 1.2. We prove the theorem according to the various steps described in Section 3. By Lemma 2.4, Steps 1 and 2, which construct the G_α-orbits with probability at least $1 - \varepsilon/3$, can be done in

$$O((\xi + N\log N)\log N(\log N + \log(3\varepsilon^{-1})))$$

time. Next, by Lemma 4.2, Step 3, the construction of the first jellyfish $\mathbf{J}_1$ and identifying set $\mathbf{I}_1$, can be done with probability at least $1 - \varepsilon/3$ in

$$O(N\log N(\log N + \log(3\varepsilon^{-1})))$$

time. The identifying set constructed has size $|\mathbf{I}_1| = \lceil\log(3N/\varepsilon)\rceil$. Steps 4 and 5 involve constructing the set Σ of all jellyfish, as well as the Tables $T1$, $T2$, $T3$ and $T4$, and verifying that Σ is G-invariant. These steps can be done in

$$O(|X|N\log N(\log N + |\mathbf{I}_1|)) = O(|X|N\log N(\log N + \log(3\varepsilon^{-1})))$$

time, by Lemma 5.2. Step 6, in which we construct the identifying pairs, can be done, by Lemma 6.1, with probability at least $1 - \varepsilon/3$ in

$$O(N + \log N\log(3\varepsilon^{-1}))$$

time. Finally Step 7, computing the images of the $|X|$ generators as permutations of Σ, can be done in

$$O(|X|nr^2k\log(rk)) = o(|X|N)$$

time, by Lemma 6.2.

Drawing together these estimates, we see first that the probability the algorithm will succeed is at least $1 - \varepsilon$. The cost is at most

$$O((\xi + |X|N + N\log N)\log N(\log N + \log(\varepsilon^{-1}))).$$

At this point we have completed tasks (i) and (ii) of Definition 1.1 for identifying G constructively. The deterministic algorithm to compute the image of any given $g \in G$ as a permutation of $\{1, \ldots, nr\}$ is given in Lemma 6.2, and the time per element is

$$O(nr^2k\log(rk)) = o(N).$$

This completes the proof.

https://doi.org/10.1112/S1461157000001236 Published online by Cambridge University Press

References

1. László Babai, Gene Cooperman, Larry Finkelstein and Ákos Seress, 'Nearly linear time algorithms for permutation groups with a small base', *Proc. International Symposium on Symbolic and Algebraic Computation,* ISSAC '91 (ed. S. M. Watt, ACM Press, New York, 1991) 200–209. 160
2. László Babai, Eugene M. Luks and Ákos Seress, 'Fast management of permutation groups', *Proc. 29th IEEE Symposium on Foundations of Computer Science*, FOCS'88 (IEEE Computer Society Press, Los Alamitos, 1988) 272–282. 160, 161
3. László Babai, Eugene M. Luks and Ákos Seress, 'Fast management of permutation groups I', *SIAM J. Computing* 26 (1997) 1310–1342. 160, 161, 165
4. Gene Cooperman, Larry Finkelstein and Namita Sarawagi, 'A random base change algorithm for permutation groups', *Proc. International Symposium on Symbolic and Algebraic Computation,* ISSAC '90 (ed. S. Watanabe and M. Nagata, ACM Press, New York, 1990) 161–168. 163
5. Peter J. Cameron, 'Finite permutation groups and finite simple groups', *Bull. London Math. Soc.* 13 (1981) 1–22. 160
6. John D. Dixon and Brian Mortimer, *Permutation groups*, Graduate Texts in Math. 163 (Springer, New York, 1996). 163
7. The GAP Group, 'GAP – Groups, Algorithms, and Programming', Version 4.4, 2005, http://www.gap-system.org. 160
8. Max Neunhöffer and Ákos Seress, 'A data structure for a uniform approach to computations with finite groups', *Proc. International Symposium on Symbolic and Algebraic Computation,* ISSAC '06 (ACM Press, New York, 2006), to appear. 160
9. Ákos Seress, *Permutation group algorithms*, Cambridge Tracts in Mathematics 152 (Cambridge University Press, Cambridge, 2003). 160, 163, 164, 165
10. Charles C. Sims, 'Computation with permutation groups', *Proc. Second Symposium on Symbolic and Algebraic Manipulation*, SYMSAC '71 (ed. S. R. Petrick, ACM Press, New York, 1971) 23–28. 159

Maska Law maska@maths.uwa.edu.au
Alice C. Niemeyer alice@maths.uwa.edu.au
Cheryl E. Praeger praeger@maths.uwa.edu.au

School of Mathematics and Statistics
The University of Western Australia
35 Stirling Highway
Crawley
Australia 6009

Ákos Seress akos@math.ohio-state.edu

Department of Mathematics
The Ohio State University
Columbus Ohio 43210
USA

https://doi.org/10.1112/S1461157000001236 Published online by Cambridge University Press