Computing normalisers of intransitive groups

Mun See Chang\textsuperscript{a,}\textsuperscript{*}, Christopher Jefferson\textsuperscript{a},
Colva M. Roney-Dougal\textsuperscript{b}

\textsuperscript{a} School of Computer Science, University of St Andrews, St Andrews, KY16 9SX, United Kingdom
\textsuperscript{b} School of Mathematics and Statistics, University of St Andrews, St Andrews, KY16 9SS, United Kingdom

\section*{Article Info}

Article history:
Received 1 December 2021
Available online 10 May 2022
Communicated by Derek Holt

Keywords:
Computational group theory
Permutation groups
Backtrack search

\section*{Abstract}
The normaliser problem takes as input subgroups $G$ and $H$ of the symmetric group $S_n$, and asks one to compute $N_G(H)$. The fastest known algorithm for this problem is simply exponential, whilst more efficient algorithms are known for restricted classes of groups. In this paper, we will focus on groups with many orbits. We give a new algorithm for the normaliser problem for these groups that performs many orders of magnitude faster than previous implementations in \textsc{GAP}. We also prove that the normaliser problem for the special case $G = S_n$ is at least as hard as computing the group of monomial automorphisms of a linear code over any field of fixed prime order.

\textcopyright 2022 The Author(s). Published by Elsevier Inc. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

\section*{1. Introduction}

Given generators for subgroups $G$ and $H$ of the symmetric group $S_n$, the normaliser problem asks one to compute a generating set for the normaliser $N_G(H)$. The fastest
known bound for this problem, in general, is Wiebking’s simply exponential bound $2^{O(n)}$ [20]. Better bounds are known for various cases: for example, quasipolynomial $2^{O(\log^3 n)}$ if $N_{S_n}(H)$ is primitive [16,5], and polynomial if $G$ has restricted composition factors, by work of Luks and Miyazaki [13]. In terms of practical computation, there are many algorithms to solve special cases of the normaliser problem (see [8,11,14] for examples).

As a consequence of Babai’s quasipolynomial solution to the string isomorphism problem [1], the intersection of permutation groups can be computed in quasipolynomial time. So, with a quasipolynomial cost, it suffices to compute $N_{S_n}(H)$, then $N_G(H) = N_{S_n}(H) \cap G$. Furthermore, in practice, computing intersections is much faster than computing normalisers. We shall therefore focus on the following problem.

**Problem 1.1 (Norm-Sym).** Given $H = \langle X \rangle \leq S_n$, compute $N_{S_n}(H)$.

To better understand worst-case complexity, it is helpful to study the case where a problem seems to be the hardest. In this paper, we will consider intransitive groups, for which Norm-Sym is not known to be solvable in quasipolynomial time. Since $N_{S_n}(H)$ may only permute permutation isomorphic $H$-orbits, the following case seems likely to be hardest.

**Definition 1.2.** Let $A \leq S_m$ be transitive. Let $\mathfrak{I}(A)$ consist of all groups $H$ for which there exists $k \geq 1$ such that $H$ is a subgroup of $S_{mk}$ with orbits $\Omega = \{\Omega_1, \Omega_2, \ldots, \Omega_k\}$, and each restriction $H|\Omega_i$ is permutation isomorphic to $A$.

In particular, we will consider $\mathfrak{I}(A)$ with $A$ the regular representation of the cyclic group $C_p$ or the natural representation of the dihedral group $D_{2q}$ of order $2q$, where $p$ and $q$ are primes and $q$ is odd.

Luks proved that the graph isomorphism problem is polynomial-time reducible to Norm-Sym [12], which is a special case of the normaliser problem. However, not much more is known about where Norm-Sym fits into the complexity hierarchy. In particular, we do not know its relation to the intersection problem, or whether it is polynomial-time equivalent to the general normaliser problem. In 1993, Luks asked [12] for a decision problem polynomial-time equivalent to Norm-Sym, but this remains open. Notably, we do not know if Conj-Sym, the problem of deciding if subgroups $H_1$ and $H_2$ of $S_n$ are conjugate in $S_n$, is polynomial-time equivalent to Norm-Sym.

Let MAut be the problem of computing the monomial automorphism group of a linear code $C$, and let MEQ be the problem of deciding if linear codes $C_1$ and $C_2$ are monomially equivalent (see Definition 3.4 for details). Petrank and Roth showed that MEQ for codes of length $k$ over $F_p$ is at least as hard as the graph isomorphism problem [15], and MEQ has time complexity bounded by $2^{O((p-1)k)}$, assuming constant time field operations [17,2]. Our first main complexity result is the following.

**Theorem 1.3.** For a fixed prime $p$, Norm-Sym and Conj-Sym for groups in $\mathfrak{I}(C_p)$ are polynomial-time equivalent to MAut and MEQ, respectively, for codes over $F_p$. 
As a consequence, NORM-SYM is at least as hard as MAUT for codes over any fixed field of prime order (see Corollary 3.6).

As Wiebking’s simply exponential algorithm [20] is not implemented, the fastest implemented algorithm to compute $N_{S_n}(H)$ has a runtime complexity of $2^{O(n \log n)}$. Using methods based on the work of Feulner [6], we shall bound the complexity of NORM-SYM for $H \in \mathcal{InP}(C_p)$ by $2^{O(\frac{n}{p} \log \frac{n}{p} + \log n)}$. We show that we may also bound the depth of the search tree (see Section 2.1) in $S_n$ by $n/(2p)$, using dual codes. Our second main complexity result will then follow immediately from Proposition 4.2 and Theorem 4.10.

**Theorem 1.4.** The NORM-SYM problem for $H$ in class $\mathcal{InP}(C_p)$ can be solved in time
\[
\min \left( 2^{O(\frac{n}{p} \log \frac{n}{p} + \log n)} , 2^{O(\frac{n}{2p} \log n)} \right).
\]

Current practical methods to solve NORM-SYM for groups in $\mathcal{InP}(C_p)$ are very slow. The implementation in the computer algebra system GAP [7] struggles to compute $N_{S_n}(H)$ even when $H$ has very small degree (median time of more than 10 minutes for degree around 30, see Section 8). This paper also develops an effective practical algorithm to solve NORM-SYM for groups in $\mathcal{InP}(C_p)$, using the above ideas. In Section 8, we provide evidence that our new algorithm performs far better than the one currently implemented in GAP.

The structure of this paper is as follows. In Section 2, we give some background and present general results on computing $N_{S_n}(H)$ for intransitive groups $H$. In Section 3, we prove Theorem 1.3 and in Section 4, we shall prove Theorem 1.4. In Section 5, we present several techniques for speeding up a backtrack search to compute $N_{S_n}(H)$ for $H \in \mathcal{InP}(C_p)$. We will describe our algorithm to compute $N_{S_n}(H)$ for such $H$ in Section 6. In Section 7, we extend our methods to class $\mathcal{InP}(D_{2q})$, where $q$ is an odd prime. Finally, in Section 8, we present runtime data.

### 2. Background and preliminaries

In this section, we consider NORM-SYM for an arbitrary intransitive group $H \leq S_n$ with orbits $\mathcal{O} = \{ \Omega_1, \Omega_2, \ldots, \Omega_k \}$. In Section 2.1, we give some preliminary complexity results and present a short overview of normaliser computation using backtrack search. In Section 2.2, we discuss equivalent orbits, then in Section 2.3, for a transitive group $A$, we construct a natural subgroup $L$ of $S_n$ containing $N_{S_n}(H)$ for $H \in \mathcal{InP}(A)$.

#### 2.1. Permutation group algorithms

In this section, let $G \leq S_n$ be given by a generating set $X$ and let $\Omega = \{1, 2, \ldots, n\}$. It is well known that in time $O(|X|n^2 + n^5)$, we may replace $X$ by a generating set of size at most $n$ (see, for example, [18]). Therefore, we assume that $|X| \leq n$ and we measure the complexity of permutation group problems in terms of $n$. 
A base $B$ of $G$ is a tuple $(\beta_1, \beta_2, \ldots, \beta_m) \in \Omega^m$ such that the pointwise stabiliser $G_{(\beta_1, \beta_2, \ldots, \beta_m)}$ is trivial. The base image of an element $g$ of $G$ relative to the base $B$ is $B^g := (\beta_1^g, \beta_2^g, \ldots, \beta_m^g)$.

Given generating sets $\{z_1, z_2, \ldots, z_k\}$ and $\{y_1, y_2, \ldots, y_l\}$ for $G \leq S_n$ and $K \leq S_m$ respectively, along with a subset $\{x_1, x_2, \ldots, x_k\}$ of $S_m$, we can check if the map $\phi$ defined by $\phi(z_i) = x_i$ for $1 \leq i \leq k$ extends to a homomorphism $\phi : G \to K$ in polynomial time [18]. We say that a homomorphism $\phi : G \to K$ is given by generator images if it is encoded by a list $[z_1, \ldots, z_k, y_1, \ldots, y_l, \phi(z_1), \ldots, \phi(z_k)]$. We shall assume that all homomorphisms are given by generator images.

The following results are standard and we refer the reader to [9,18] for further details.

**Lemma 2.1.** Let $G = \langle X \rangle \leq S_n$.

1. In polynomial time, we can: compute the orbits of $G$; compute the order $|G|$; compute the restriction $G|_\Delta$ of $G$ to a given $G$-invariant set $\Delta$; compute a base for $G$; compute the centraliser $C_{S_n}(G)$.
2. Given a permutation group $K$, a homomorphism $\phi : G \to K$, and permutations $g \in G$ and $k \in \text{Im}(\phi)$, in polynomial time we can compute the image $\phi(g)$ and a pre-image of $k$.

We compute normalisers using backtrack search (see, for example, [9,18] for more details). Suppose that we want to compute $N_G(H)$. The search tree $T$ of $G$ with respect to a base $B$ is a rooted tree of depth $m$, where the root node is labelled by the empty tuple $()$ and nodes at depth $d$ are labelled with elements of the orbit $(\beta_1, \beta_2, \ldots, \beta_d)^G$ such that each node $(\alpha_1, \alpha_2, \ldots, \alpha_d)$ has parent $(\alpha_1, \alpha_2, \ldots, \alpha_{d-1})$. We can associate each node of $T$ with a coset of a point stabiliser in $G$ by defining

$$\Psi : (\alpha_1, \alpha_2, \ldots, \alpha_d) \mapsto \{g \in G \mid \beta_1^g = \alpha_i, \text{ for all } i \in \{1, 2, \ldots, d\}\}.$$  

We traverse the search tree $T$ depth-first and gather the elements of $N_G(H)$ that we find in a group $N$, which is updated as search progresses.

When solving problems with backtrack search, the runtime is correlated to the number of nodes in the search tree we visit. We reduce this number in two ways. Firstly, if there exists a proper subgroup $S$ of $G$ containing $N_G(H)$, we search in the tree of $S$ with respect to a base of $S$ instead. We will describe how we can find such an $S$ for $H \in \mathfrak{InP}(C_p) \cup \mathfrak{InP}(D_{2p})$ in Section 2.3. Secondly, if we can deduce for some node $\tau$ of $T$ that $\Psi(\tau) \cap N_G(H)$ is empty, or that all elements of $\Psi(\tau) \cap N_G(H)$ are contained in the group $N$ of normalising elements we have already found, then we can skip traversing the subtree rooted at $\tau$. We call this skipping pruning $T$.

Methods for deducing that $\Psi(\tau) \cap N_G(H)$ is a non-empty subset of $N$ are known, see for example [18, Section 9.1], but we present new, efficient, techniques for deducing that
\( \Psi(\tau) \cap N_G(H) = \emptyset \) for \( H \in \mathfrak{m}(C_p) \) in Sections 4 and 5. We will also use the following elementary lemma (whose proof is clear) to deduce that \( \Psi(\tau) \cap N_G(H) = \emptyset \).

**Lemma 2.2.**

1. If there exists \( \sigma \in N_{S_n}(H) \) such that \((\delta_1, \delta_2, \ldots, \delta_m)\sigma = (\gamma_1, \gamma_2, \ldots, \gamma_m)\), then \( (H(\delta_1, \delta_2, \ldots, \delta_m))\sigma = H(\gamma_1, \gamma_2, \ldots, \gamma_m) \).

2. Let \( \Delta_1 \) and \( \Delta_2 \) be unions of \( H \)-orbits. If there exists \( \sigma \in N_{S_n}(H) \) such that \( \Delta_1^\sigma = \Delta_2 \), then \( (H|_{\Delta_1})\sigma = H|_{\Delta_2} \).

Throughout the paper, for all subsets \( \Delta \) of \( \Omega \), we shall consider subgroups of \( \text{Sym}(\Delta) \) as subgroups of \( \text{Sym}(\Omega) \) with support \( \Delta \).

### 2.2. Equivalence of orbits

In this subsection, we define an equivalence relation on the set \( \mathcal{O} \) of \( H \)-orbits, and show how it is used in centraliser and normaliser computation.

**Definition 2.3.** Two orbits \( \Omega_i \) and \( \Omega_j \) of \( H \) are equivalent, denoted \( \Omega_i \equiv_H \Omega_j \), if there exists a bijection \( \psi : \Omega_i \to \Omega_j \) such that

\[
\psi(\delta^h) = \psi(\delta)^h \quad \text{for all } h \in H \text{ and } \delta \in \Omega_i.
\]

We say that \( \psi \) witnesses the equivalence.

For \( \Omega_i \) and \( \Omega_j \) in \( \mathcal{O} \) and a bijection \( \varphi : \Omega_i \to \Omega_j \), we denote by \( \overline{\varphi} \) the involution in \( \text{Sym}(\Omega_i \cup \Omega_j) \) such that \( \alpha \overline{\varphi} = \varphi(\alpha) \) for all \( \alpha \in \Omega_i \). Hence \( \Omega_i \) and \( \Omega_j \) are equivalent if and only if there exists an involution \( \sigma = \overline{\psi} \in \text{Sym}(\Omega_i \cup \Omega_j) \) such that

\[
h|_{\Omega_j} = (h|_{\Omega_i})\sigma \quad \text{for all } h \in H.
\]

Next we see how the relation \( \equiv_H \) helps us compute \( C_{S_n}(H) \) for intransitive groups \( H \).

**Lemma 2.4** ([18, §6.1.2]). Let \( H \leq S_n \) and let \( \mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_a \) be the \( \equiv_H \)-classes. For \( 1 \leq i \leq a \), let \( \mathcal{B}_i := \{ \Omega_{i1}, \Omega_{i2}, \ldots, \Omega_{|\mathcal{B}_i|} \} \), and for \( 2 \leq j \leq |\mathcal{B}_i| \), let \( \psi_{ij} : \Omega_{i1} \to \Omega_{ij} \) witness the equivalence. Let \( \mathcal{B}_i := \langle \overline{\psi_{ij}} \mid 2 \leq j \leq |\mathcal{B}_i| \rangle \) and let \( C_i := C_{\text{Sym}(\Omega_{i1})}(H|_{\Omega_{i1}}) \).

Then

\[
C_{S_n}(H) = \langle C_1 \times C_2 \times \cdots \times C_k, B_1 \times B_2 \times \cdots \times B_a \rangle \cong \prod_{i=1}^{a} C_{\text{Sym}(\Omega_{i1})}(H|_{\Omega_{i1}}) \wr S_{|\mathcal{B}_i|}.
\]

In particular, the elements \( \overline{\psi_{ij}} \), and \( C_{S_n}(H) \), can be computed in polynomial time.
By Lemma 2.2, we can deduce that there are no normalising elements in a subtree of $T$ by showing that some restrictions or stabilisers are not conjugate in $S_n$. We can use orbit equivalence to show that subgroups of $S_n$ are not conjugate.

Lemma 2.5. Let $R$ and $Q$ be subgroups of $S_n$ such that $Q = R^\sigma$ for some $\sigma \in S_n$, and let $\mathcal{B}$ be an $\equiv_R$-class of orbits. Then $\{\Delta^\sigma \mid \Delta \in \mathcal{B}\}$ is an $\equiv_Q$-class. Hence, with $H$ as in Lemma 2.4, the group $N_{S_n}(H)$ acts on $\{\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_a\}$.

Proof. If $\Delta$ and $\Gamma$ are equivalent $R$-orbits, then by (2) there exists an involution $\mu$ in $\text{Sym}(\Delta \cup \Gamma)$ such that $r|_\Gamma = (r|_\Delta)^\mu$ for all $r \in R$. Let $q \in Q$. Then there exists $r \in R$ such that $r^\sigma = q$. So

$$q|_\Gamma^\sigma = (r|_\Gamma)^\sigma = (r|_\Delta)^{\mu^\sigma} = (r|_\Delta)^{\sigma^{-1}\mu^\sigma} = (q|_\Delta)^{\sigma^{-1}\mu^\sigma},$$

and hence $\Delta^\sigma \equiv_Q \Gamma^\sigma$ by (2). □

Next, we show that if $\equiv_H$ is not the equality relation then the computation of $N_{S_n}(H)$ reduces in polynomial time to computing the normaliser of a group $H_1$ of a smaller degree for which $\equiv_{H_1}$ is equality. For a partition $P$ of a set $S$, we denote by $[s]$, or $[s]_P$ if we wish to emphasise $P$, the cell of $P$ containing $s$. For $s, s' \in S$, we write $s \sim s'$ (or $s \sim_P s'$) to mean that $s$ and $s'$ are in the same cell of $P$, so $[s] = [s']$.

Proposition 2.6. NORM-SYM for $H \leq S_n$ reduces in polynomial time to the special case where the $H$-orbits are pairwise inequivalent.

Proof. We may assume that $H$ is intransitive and that $\equiv_H$ is not the equality relation. With the notation of Lemma 2.4, denote by $\mathcal{P}(H)$ the partition of $\{\Omega_{11}, \Omega_{21}, \ldots, \Omega_{11}\}$ such that $\Omega_{11} \sim \Omega_{11}$ if and only if $|\mathcal{B}_i| = |\mathcal{B}_j|$. Let $\Gamma := \bigcup_{i=1}^k \Omega_{11}$ and let $U$ be the stabiliser of $\mathcal{P}(H)$ in $\text{Sym}(\Gamma)$. We prove the claim by showing that, in polynomial time, one can construct a homomorphism $\theta : U \rightarrow S_n$ such that $N_{S_n}(H) = \langle \theta(N_U(H|_\Gamma)), C_{S_n}(H) \rangle$.

Let the bijections $\psi_{ij} : \Omega_{11} \rightarrow \Omega_{11}$ be as in Lemma 2.4 and let $\psi_{11} = 1$. For all $u \in U$, define $\theta(u)$ by: for all $\alpha \in \Omega_{is}$, if $\Omega_{i1}^u = \Omega_{j1}$ then $\alpha^{\theta(u)} = \alpha^{\psi_{is}^{-1}\psi_{js}} \in \Omega_{js}$. One can check that $\theta(u)$ is indeed in $S_n$ and that $\theta$ is a homomorphism.

To see that $N_{S_n}(H)$ contains $\langle \theta(N_U(H|_\Gamma)), C_{S_n}(H) \rangle$, it suffices to show that it contains the image of $N_U(H|_\Gamma)$ under $\theta$. Let $h \in H$ and $u \in N_U(H|_\Gamma)$. Then there exists an $h' \in H$ such that $h'|_\Gamma = (h|_\Gamma)^u$. As $\psi_{j1} = 1$, for all $\alpha \in \Gamma$, the image $\alpha^{\theta(u)} = \alpha^u$ is in $\Gamma$. So $h^{\theta(u)}|_\Gamma = (h|_\Gamma)^u = h'|_\Gamma$. Now for $1 \leq i \leq t$ and $2 \leq s \leq |\mathcal{B}_i|$, by the definition of $\psi_{is}$,

$$h^{\theta(u)}|_{\Omega_{is}} = (h^{\theta(u)}|_{\Omega_{i1}})^{\psi_{is}} = (h'|_{\Omega_{i1}})^{\psi_{is}} = h'|_{\Omega_{is}},$$

so $h^{\theta(u)} = h'$. Hence $\theta(u) \in N_{S_n}(H)$.

To show the converse containment, let $\nu \in N_{S_n}(H)$. By Lemma 2.5, $\nu$ acts on the $\equiv_H$-classes $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_a$. By Lemma 2.4, $C_{S_n}(H)$ induces the full symmetric group,
independently, on each $B_i$. So there exists a $c \in C_{S_n}(H)$ such that $\nu c$ fixes $\Gamma$ setwise, and so $(\nu c)|_{\Gamma} \in N_U(H|_{\Gamma})$. Let $\sigma := \theta((\nu c)|_{\Gamma})$ and $h \in H$. Then $(h|_{\Gamma})^{\nu c} = (h|_{\Gamma})^{\sigma}$ and so by the argument of the previous paragraph $h^{\nu c} = h^{\sigma}$. Therefore $\nu \sigma^{-1} \in C_{S_n}(H)$ and hence $\nu \in \langle \theta(N_U(H|_{\Gamma})), C_{S_n}(H) \rangle$.

Since the permutations $\psi_{ij}$ and $C_{S_n}(H)$ can be computed in polynomial time by Lemma 2.4, the complexity claim is immediate. □

Hence in much of the remainder of the paper, we shall assume that the $\equiv_H$-relation is the equality relation. That is, we assume that the $H$-orbits are pairwise inequivalent.

2.3. Normalisers of groups in $\mathfrak{InP}(A)$: overgroups

We now assume that there exists an integer $m$ and a transitive subgroup $A$ of $S_m$ such that $H$ is in $\mathfrak{InP}(A)$.

**Definition 2.7.** A subgroup $R$ of $S = Q_1 \times Q_2 \times \cdots \times Q_k$ is a subdirect product of $S$ if each projection of $R$ onto $Q_i$ is surjective.

Let $G_i := H|_{\Omega_i}$ for $1 \leq i \leq k$, where as before the $\Omega_i$ are the orbits of $H$. Then $H$ is a subdirect product of $G = G_1 \times G_2 \times \cdots \times G_k$, where we identify the direct product with the corresponding subgroup of $S_n$. We call $G$ the enveloping group of $H$.

Two permutation groups $R \leq \text{Sym}(\Delta)$ and $S \leq \text{Sym}(\Gamma)$ are permutation isomorphic if there exists a bijection $\phi : \Delta \to \Gamma$ and an isomorphism $\psi : R \to S$ such that

$$
\phi(\delta^r) = \phi(\delta)^{\psi(r)}
$$

for all $\delta \in \Delta$ and $r \in R$. We say that $\phi$ witnesses the permutation isomorphism from $R$ to $S$.

Next we define a subgroup $L \leq S_n$, analyse its structure and show that $L$ contains $N_{S_n}(H)$.

**Lemma 2.8.** Let $G = G_1 \times G_2 \times \cdots \times G_k$ be the enveloping group of $H$. For $2 \leq j \leq k$, let $\phi_j : \Omega_1 \to \Omega_j$ witness the permutation isomorphism from $G_1$ to $G_j$. For $1 \leq i \leq k$, let $N_i := N_{\text{Sym}(\Omega_i)}(G_i)$. Let $B := \langle N_1, N_2, \ldots, N_k \rangle \leq S_n$, let $K := \langle \phi_j \mid 2 \leq j \leq k \rangle \leq S_n$, and let $L := \langle B, K \rangle$. Then

1. $K$ acts faithfully as $\text{Sym}(\Omega)$ on the set $\mathcal{O}$ of $H$-orbits;
2. $L$ is permutation isomorphic to $N_1 \wr S_k$ in its imprimitive action;
3. $N_{S_n}(H) \leq L$;
4. in polynomial time, given $l \in L$, we can compute $b \in B$ and $\kappa \in K$ such that $l = bk$.

**Proof.** Part 1: Let $\xi : K \to S_k$ be the permutation representation of $K$ on $\mathcal{O}$. Then $\xi$ is surjective since $\bar{\phi_i}$ induces the permutation $(\Omega_1, \Omega_i)$ on $\mathcal{O}$. For injectivity, let $\kappa \in K$
be such that $\xi(\kappa) = 1$. Then $\kappa$ setwise stabilises each $\Omega_i$. Fix $i$, let $\alpha \in \Omega_i$, and consider $\kappa$ as a word in the $\overline{\phi}_j$. Since $\Omega_i$ is moved only by $\overline{\phi}_i$, if $\overline{\phi}_i$ does not occur in $\kappa$ then $\alpha$ is fixed by $\kappa$, so assume otherwise. As $\alpha^\kappa$ is also in $\Omega_i$, there exists a subword $\kappa'$ of $\kappa$ such that $\alpha^\kappa = \alpha^{\overline{\phi}_i \kappa \overline{\phi}_i}$, where $\alpha^{\overline{\phi}_i}$ and $\alpha^{\overline{\phi}_i \kappa'}$ are in $\Omega_i$. Now since each $\Omega_j$ is moved only by $\overline{\phi}_j$, there exist (not necessarily distinct) integers $l_1, l_2, \ldots, l_r \in \{2, 3, \ldots, k\}$ such that $(\alpha^{\overline{\phi}_i})^{\kappa'} = (\alpha^{\overline{\phi}_i})^{\overline{\phi}_{l_1} \overline{\phi}_{l_2} \cdots \overline{\phi}_{l_r}} = \alpha^{\overline{\phi}_i}$. Therefore $\alpha^\kappa = \alpha$.

Part 2: We first show that $L$ is imprimitive with block system $O$. Let $\alpha \in \Omega_i$ and $\beta \in \Omega_j$. Then $\alpha^{\overline{\phi}_j}$ and $\beta^{\overline{\phi}_j}$ are points in $\Omega_1$, so there exists $g \in G \leq L$ such that $\alpha^{\overline{\phi}_j}g = \beta^{\overline{\phi}_j}$, and hence $L$ is transitive. Since $B$ and $K$ preserve $O$, it follows that $O$ is a block system for $L$.

Consider the kernel $J$ of the action of $L$ on $O$. Since the action of $K$ on $O$ is faithful, $J \leq B$. Conversely, since each $N_i$ fixes $O$, the group $B$ is a subgroup of $J$. So $J$ is isomorphic to $N_1 \times N_2 \times \cdots \times N_k$, and hence, up to isomorphism, $L \leq N_1 \wr S_k$. But as $S_k \cong K \leq L$, it follows that $L \cong N_1 \wr S_k$.

Part 3: The normaliser $N_{S_n}(H)$ is permutation isomorphic to a subgroup of $N_1 \wr S_k$ in its natural imprimitive action [10, \S 11], so this follows from Part 2.

Part 4: The element $l$ induces a permutation $\sigma$ of the set $O$ of orbits. Then $\kappa = \xi^{-1}(\sigma)$ can be computed in polynomial time by Lemma 2.1.2, and $l\kappa^{-1}$ fixes each $H$-orbit, so is in $B$.  

3. $\mathcal{Jn}\Psi(C_p)$ and automorphisms of codes

In this section, let $p$ be prime. For the rest of this section, we shall assume that the following hypothesis holds.

**Hypothesis 3.1.** Let $n = pk$ and let $H$ be a subgroup of $S_n$ in class $\mathcal{Jn}\Psi(C_p)$. Let the set $O := \{\Omega_1, \Omega_2, \ldots, \Omega_k\}$ be the orbits of $H$, ordered such that $|H| = |(H|_{\cup_i \Omega_i})| = p^s$ for some $s$.

Let $G$ be the enveloping group of $H$ and let $g_i$ be a permutation in $\text{Sym}(\Omega_i)$ generating $G_i$. For $2 \leq i \leq k$, let $\phi_i : \Omega_1 \to \Omega_i$ witness the permutation isomorphism between $G_1$ and $G_i$, and let $g_i = g_1^{\phi_i}$.

We now set up an isomorphism $\gamma$ from $H$ to a linear code. Then, in Section 3.1, we shall prove that computing $N_{S_n}(H)$ is polynomial-time equivalent to computing the monomial automorphism group of $\gamma(H)$.

Denote the set of all $s \times k$ matrices over the field $\mathbb{F}_p$ by $M(s, k, p)$. For $M \in M(s, k, p)$, we denote by $M_i, s$ and $M_i, j$ the $i$-th row and the $j$-th column of $M$ respectively. For a tuple $I$ of distinct elements of $\{1, 2, \ldots, s\}$, we denote by $M_{I, s}$ the matrix of dimension $|I| \times k$ such that the $i$-th row of $M_{I, s}$ is $M_{I, i}$, and similarly for columns. We denote the row space of $M$ by $\langle M \rangle$.

A linear code $C$ over $\mathbb{F}_p$ is a subspace of $\mathbb{F}_p^k$ for some $k$, denoted by $C \leq \mathbb{F}_p^k$. A generator matrix $M$ of $C \leq \mathbb{F}_p^k$ is a matrix in $M(s, k, p)$ for some $s$ whose rows form a
basis for \( C \). We shall assume that all linear codes are given by generator matrices. The matrix \( M \) is in \textit{standard form} if its first \( s \) columns form the identity matrix \( I_s \). For more information on linear codes, refer to, for example, [19]. Since throughout the paper we shall be moving between exponential and additive notation, we shall identify elements of \( \mathbb{F}_p \) with integers \( \{0, 1, \ldots, p - 1\} \). We denote by \( \mathbb{F}_p^\times \) the multiplicative group of \( \mathbb{F}_p \).

**Definition 3.2.** Let \( \gamma \) be the isomorphism from \( G \) to \( \mathbb{F}_p^k \) defined by

\[
\gamma(g_1^{r_1}g_2^{r_2} \cdots g_k^{r_k}) = (r_1, r_2, \ldots, r_k).
\]

Observe that \( \gamma(H) \) is a subspace of \( \gamma(G) = \mathbb{F}_p^k \), so \( \gamma(H) \) is linear code of length \( k \) and dimension \( s \) over \( \mathbb{F}_p \). Our assumption that \( |\{H|_{\cup_{i=1}^s \Omega_i}\}| = p^s \) means that no column permutations are needed to put a generator matrix of \( \gamma(H) \) in standard form, so let \( M \) be such a generator matrix.

We show that, in polynomial time, we can decide if a given subgroup of \( S_n \) is in \( \text{InP}(\mathbb{C}_p) \), and if so compute some accompanying data.

**Lemma 3.3.** Let \( Q = \langle Y \rangle \) be a subgroup of \( S_n \) with \( k \) orbits. Then in polynomial time, we can decide if \( Q \in \text{InP}(\mathbb{C}_p) \). Furthermore, for \( H \) as in Hypothesis 3.1, in polynomial time, we can compute: a suitable ordering \( \Omega_1, \Omega_2, \ldots, \Omega_k \) of \( O \), the bijections \( \phi_i \) and the generators \( g_i \) from Hypothesis 3.1; the isomorphism \( \gamma \) from Definition 3.2; and a generator matrix \( M \in M(s, k, p) \) for \( \gamma(H) \) in standard form.

**Proof.** The group \( Q \) is in \( \text{InP}(\mathbb{C}_p) \) if and only if for all \( Q \)-orbits \( \Delta \), the size of \( \Delta \) is \( p \) and \( |\langle Q|_{\Delta}\rangle| = p \). So we can decide if \( Q \in \text{InP}(\mathbb{C}_p) \) in polynomial time by Lemma 2.1. The ordering of the \( H \)-orbits can be obtained in polynomial time by Lemma 2.1.1. Choices for \( g_i \) can be computed in polynomial time, since any non-trivial restriction \( x_{|\Delta} \) of \( x \in X \) generates \( H_{|\Delta} \). Since conjugacy of permutations in the symmetric group can be computed in polynomial time, so can the bijections \( \phi_i \). We construct \( \gamma \) by mapping each \( g_i \) to the \( i \)-th standard basis vector of \( \mathbb{F}_p^k \). Finally, \( M \) can be obtained by finding a row reduced basis for \( \langle \gamma(x) \mid x \in X \rangle \), in time polynomial in \( n \in O(n) \) and \( k \in O(n) \) [3].

3.1. The normaliser of \( H \) as an automorphism group of a linear code

In this subsection we prove Theorem 1.3, but first we define the actions of certain subgroups of \( \text{GL}_k(p) \) on \( \gamma(H) \).

Let \( D \) and \( P \) be the groups of all diagonal and permutation matrices in \( \text{GL}_k(p) \), respectively, and let \( W := \langle D, P \rangle \). The natural action of \( W \) on \( \mathbb{F}_p^k \) is called the \textit{monomial action}. Observe that \( D \cong (\mathbb{F}_p^\times)^k \), \( P \cong S_k \) and \( W = D \rtimes P \cong \mathbb{F}_p^\times \wr S_k \).

**Definition 3.4.** Let \( C \leq \mathbb{F}_p^k \) be a linear code. The \textit{monomial automorphism group} \( \text{MAut}(C) \) of \( C \) is the subgroup of \( W \) that setwise stabilises \( C \). Two codes \( C, C' \leq \mathbb{F}_p^k \) are
monomially equivalent if \( Cw = C' \) for some \( w \in W \) (we denote vector-matrix multiplication by concatenation).

Let \( L = B \times K \) be as in Lemma 2.8. Observe that \( N_{S_p}(C_p) = \text{AGL}_1(p) \cong C_p \times C_{p-1} \), so \( L \cong (C_p \times C_{p-1}) \wr S_k \). We shall show that the conjugation action of \( L / G \cong C_{p-1} \wr S_k \) on \( G \) is equivalent to the action of \( W \) on \( F_p^k \), and hence show that computing \( N_{S_n}(H) \) is polynomial-time equivalent to computing \( \text{MAut}(\gamma(H)) \).

**Lemma 3.5.** Define a homomorphism \( \rho: K \to P \) by \( \rho(\kappa)_{i,j} = 1 \) if and only if \( \Omega_{i}^{\kappa} = \Omega_{j} \). Let \( \zeta: B \to D \) map \( b \) to \( \text{diag}(d_1, d_2, \ldots, d_k) \) where \( b^{-1}g_i b = g_i^{d_i} \) for \( 1 \leq i \leq k \). Define a map \( \Xi: L \to W \) by writing each \( l \in L \) as \( b\kappa \) for some \( b \in B \) and \( \kappa \in K \) and mapping

\[
l = b\kappa \mapsto \zeta(b)\rho(\kappa).
\]

Then the following statements hold.

1. \( \Xi \) is an epimorphism with kernel \( G \).
2. \( \gamma(g^i) = \gamma(g)\Xi(l) \) for all \( g \in G \) and \( l \in L \).
3. \( \Xi(N_{S_n}(H)) = \text{MAut}(\gamma(H)) \), so \( N_{S_n}(H) \) is the full pre-image of \( \text{MAut}(\gamma(H)) \) under \( \Xi \).
4. Given \( l \in L \) and \( w \in W \), we can compute both \( \Xi(l) \) and a preimage of \( w \) under \( \Xi \) in time polynomial in \( n \).

**Proof.** Part 1: Let \( b_1, b_2 \in B \) and \( \kappa_1, \kappa_2 \in K \). Then

\[
\Xi(b_1 \kappa_1 b_2 \kappa_2) = \Xi(b_1 b_2^{-1} \kappa_1 \kappa_2) = \zeta(b_1)\zeta(b_2^{-1})\rho(\kappa_1)\rho(\kappa_2),
\]

so to show that \( \Xi \) is a homomorphism, it suffices to show that

\[
v\rho(\kappa_1)\zeta(b_2)\rho(\kappa_1)^{-1} = v\zeta(b_2^{-1})
\]

for all \( v \in F_p^k \). For \( i \in \{1, 2, \ldots, k\} \), let \( \Omega_j = \Omega_i^{\kappa_1} \), and let \( d_j = \zeta(b_2)_j \). Then

\[
(v\rho(\kappa_1)\zeta(b_2)\rho(\kappa_1)^{-1})_i = (v\rho(\kappa_1)\zeta(b_2))_j = (v\rho(\kappa_1))_j d_j = v_i d_j.
\]

Since \( g_i^\kappa_1 b_2 \kappa_1^{-1} = g_j^\kappa_1 b_2 \kappa_1^{-1} = g_j^d_j \kappa_1^{-1} = g_i^d_j \),

\[
(v\zeta(b_2^{-1}))_i = v_i (\zeta(b_2^{-1}))_i = v_i d_j.
\]

So \( \zeta(b_2^{-1}) = \zeta(b_2)\rho(\kappa_1)^{-1} \), and hence \( \Xi \) is a homomorphism. Since \( \zeta \) is an epimorphism, so is \( \Xi \). Finally, as \( \text{Ker}(\rho) \) is trivial, \( \text{Ker}(\Xi) = \text{Ker}(\zeta) = G \).

Part 2: Let \( l = b\kappa \) with \( b \in B \) and \( \kappa \in K \). Then there exist \( r_i \in F_p \) and \( d_i \in F_p^* \) such...
that $g|_{\Omega_i} = g^k_i$ and $g^b_i = g^{d_i}_i$. Let $\Omega_j = \Omega^k_j$. Then $(g^j)^{|\Omega_j} = ((g^j)|_{\Omega_j})^k = (g^r_i)^{d_i}$. Recall the involutions $\phi_i \in \text{Sym}(\Omega_1 \cup \Omega_2)$ from Hypothesis 3.1. Since $K$ is generated by the $\phi_i$ and conjugation by $\phi_i$ swaps $g_1$ and $g_2$, it follows that $g^k_i = g_j$. So $(g^{j}|_{\Omega_j} = g^{r_i,d_i}$ and $\gamma(g^j)_i = r_id_i$. Hence for all $j$,

$$\gamma(g)\Xi(l))_j = (\gamma(g)\zeta(b)\rho(k))_j = (\gamma(g)\zeta(b))_i = r_id_i = \gamma(g^j)_i.$$ 

Part 3: First note that $N_{S_n}(H) \leq L$ by Lemma 2.8.3. By Part 2, if $l \in L$ is in $N_{S_n}(H)$, then $\gamma(h)\Xi(l) \in \gamma(H)$ for all $h \in H$ and so $\Xi(l) \in \text{MAut}(\gamma(H))$. Conversely for an element $w$ of $\text{MAut}(\gamma(H))$, by Part 1, there exists an $l \in L$ such that $\Xi(l) = w$. Then for all $h \in H$, the element $\gamma(h) = \gamma(h)w \in \gamma(H)$. So $l \in N_{S_n}(H)$.

Part 4: By Lemma 2.8.4, in polynomial time, we can find $b \in B$ and $\kappa \in K$ such that $l = bk$. For $1 \leq i \leq k$, we can find $d_i \leq p - 1$ such that $b^{-1}g_ib = g^{d_i}_i$, and $j$ such that $\Omega^k_i = \Omega_j$. As $k \leq n$, the images $\zeta(b)$ and $\rho(\kappa)$, and hence $\Xi(l)$, can be computed in time polynomial in $n$.

To show that we can find a preimage of $w$ under $\Xi$ in polynomial time, for $1 \leq i \leq k$, let $d_i$ be the non-zero entry of $w_{i,s}$, and let $d = \text{diag}(d_1, d_2, \ldots, d_k)$. So in time polynomial in $k \log p \in O(n)$, we can compute $d \in D$ and $q := d^{-1}w \in P$ such that $w = dq$.

Now we find an element $\sigma$ of $S_n$ such that $\sigma^{-1}g_i\sigma = g^{d_i}_i$ in time polynomial in $n$, since the $g_i$ have disjoint supports and conjugation in $\text{Sym}(\Omega_i)$ can be solved in time polynomial in $|\Omega_i|$. Then $\sigma \in B$ and $\zeta(\sigma) = d \in D$. Next, in time polynomial in $k$, we construct the element $\sigma$ of $S_{k}$ such that the image $\rho^\sigma$ is the position of the non-zero entry in $q_{i,s}$. Letting $\xi$ be as in the proof of Lemma 2.8.1, $\kappa := \xi^{-1}(\sigma)$ is the element of $K$ with $\rho(\kappa) = q$, which can be computed in time polynomial in $n$ by Lemma 2.1.2. Therefore, in time polynomial in $n$, we can compute an element $\sigma\kappa$ of $L$ with $\Xi$-image $w$. \qed

Finally we prove Theorem 1.3.

Proof of Theorem 1.3. To reduce $\text{Norm-Sym}$ for groups $H \leq S_{pk}$ in class $\mathfrak{I}(C_p)$ to $\text{MAut}$, first notice that by Lemma 3.3, in polynomial time, we can compute the enveloping group $G$ of $H$ and an isomorphism $\gamma : G \to F^k_p$. Assume that we can compute a generating set $Y$ for $\text{MAut}(\gamma(H))$ in time polynomial in $k$. Then by Lemma 3.5, $N_{S_n}(H) = \langle \{\Xi^{-1}(y) \mid y \in Y\} \rangle$, where each $\Xi^{-1}(y)$ denotes a pre-image of $y$ under $\Xi$.

For the backward reduction, let $C \leq F^k_p$ be a linear code given by a generator matrix $M \in M(s, k, p)$. Let $g_i = (p(i - 1) + 1, \ldots, pi) \in S_{pk}$ for $1 \leq i \leq k$ and let the group $G = \langle g_1, g_2, \ldots, g_k \rangle$. Let $\gamma$ be as in Definition 3.2 and let $H = \langle \gamma^{-1}(M_{i,s}) \mid 1 \leq i \leq s \rangle$. Assume that we can compute a generating set $Y$ for $N_{S_{pk}}(H)$ in time polynomial in $k$. Then by Lemma 3.5.3, $\text{MAut}(C) = \Xi(N_{S_{pk}}(H)) = \langle \Xi(y) \mid y \in Y\rangle$, which can be computed in time polynomial in $k$ by Lemma 3.5.4.

For the equivalence of $\text{Conj-Sym}$ for groups in class $\mathfrak{I}(C_p)$ and $\text{MEQ}$ for codes over $F_p$, let $H_1$ and $H_2$ be subgroups of $S_{pk}$ and let $C_1 = \gamma(H_1)$ and $C_2 = \gamma(H_2)$ be codes of length $k$ over $F_p$. It follows from Lemma 3.5.1–2 that $H_1$ and $H_2$ are conjugate
in $S_n$ if and only if $C_1$ and $C_2$ are monomially equivalent. The rest of the proof is similar to that of the equivalence of $\text{NORM-SYM}$ and $\text{MAUT}$. □

**Corollary 3.6.** Fix a prime $p$. Denote $\text{MAUT}$ for $p$-ary codes by $\text{MAUT}_p$. Then the graph isomorphism problem is polynomial-time reducible to $\text{MAUT}_p$, which is polynomial-time reducible to $\text{NORM-SYM}$.

**Proof.** The graph isomorphism problem is polynomial-time reducible to $\text{MEQ}$ for $p$-ary codes [15], which is polynomial-time reducible to $\text{MAUT}_p$ by Theorem 1.3. The result now follows as $\text{NORM-SYM}$ for $\mathfrak{nP}(C_p)$ is a special case of $\text{NORM-SYM}$. □

4. Complexity results

Let $H$ be as in Hypothesis 3.1, and let $L = B \times K$ be as in Lemma 2.8. In Section 4.1, we show that $N_{S_n}(H)$ can be computed in time $2^O(N \log n)$. In Section 4.2, we show that for each $\kappa \in K$, if there exists $b \in B$ such that $b\kappa \in N_{S_n}(H)$, then we can find such a $b$ in polynomial time. In Section 4.3, we first show that $N_B(H)$ can be computed in polynomial time, then we see how the results in this section come together to reduce the search space for $N_{S_n}(H)$ to searching in $K \cong S_k$ instead of $L \cong (C_p \times C_{p-1}) \wr S_k$, and hence prove Theorem 1.4.

4.1. Limiting the depth of the search tree

In this subsection, we show how we can reduce the depth of the search tree using a possibly smaller group $H^\perp$, which we define now.

The *dual* code $C^\perp$ of a code $C \subseteq F_p^k$ is the subspace $\{v \in F_p^k \mid v \cdot c = 0 \text{ for all } c \in C\}$, where $\cdot$ denotes the standard dot product. Let $H^\perp \subseteq S_n$ be $\gamma^{-1}(\gamma(H)\perp)$. We shall show that there exists a bijection between $N_{S_n}(H)$ and $N_{S_n}(H^\perp)$. Recall from Lemma 2.8 that $N_{S_n}(H) \leq L$.

**Lemma 4.1.** Let $b \in B$ and $\kappa \in K$. Then $b\kappa \in N_{S_n}(H)$ if and only if $b^{-1}\kappa \in N_{S_n}(H^\perp)$.

**Proof.** Let the epimorphism $\Xi : L \to W$ be as in Lemma 3.5.

$\Rightarrow$: Assume that $l = b\kappa \in N_{S_n}(H)$ and let $\eta \in H^\perp$. Let $\zeta(b) = \text{diag}(d_1, d_2, \ldots, d_k) \in D$. We show that $\eta^{b^{-1}\kappa} \in H^\perp$ by showing that $\gamma(h) \cdot \gamma(\eta^{b^{-1}\kappa}) = 0$ for all $h \in H$, so let $h \in H$ and $g := h^{-1} \in H$. Then

$$0 = \gamma(g) \cdot \gamma(\eta) = \sum_{i=1}^k \gamma(g)_i \gamma(\eta)_i = \sum_{i=1}^k \gamma(g)_i d_i \gamma(\eta)_i d_i^{-1} = \gamma(g) \zeta(b) \cdot \gamma(\eta) \zeta(b^{-1}).$$

Since $P$ permutes the coordinates of $F_p^k$, the product $\gamma(g)\Xi(l) \cdot \gamma(\eta)\Xi(b^{-1}\kappa)$ is also 0. Now by Lemma 3.5.2,

$$\gamma(h) \cdot \gamma(\eta^{b^{-1}\kappa}) = \gamma(g) \cdot \gamma(\eta^{b^{-1}\kappa}) = \gamma(g)\Xi(l) \cdot \gamma(\eta)\Xi(b^{-1}\kappa) = 0.$$
\[ (H^\perp)^\perp = H \text{ implies that } b\kappa = (b^{-1})^{-1}\kappa \in N_{S_n}(H). \quad \Box \]

Now we prove a more precise version of the upper bound in Theorem 1.4: recall that we use the notation and assumptions of Hypothesis 3.1.

**Proposition 4.2.** Let \( m := \min\{s, k - s\} \). Then \( N_{S_n}(H) \) can be computed in time \( 2^{O((m+1) \log n)} \). Hence \( N_{S_n}(H) \) can be computed in time \( 2^{O(n^2 \log n)} \).

**Proof.** Notice that \( |H^\perp| = p^{k-s} \), so \( m \) is the minimum of \( \log_p(|H|) \) and \( \log_p(|H^\perp|) \). We shall show that \( N_{S_n}(H) \) can be computed in time \( 2^{O((s+1) \log n)} \). Then if \( s > k/2 \), using Lemmas 2.8.4 and 4.1, we may compute \( N_{S_n}(H) \) in time \( 2^{O((m+1) \log n)} \) by computing \( N_{S_n}(H^\perp) \) instead. From here, since \( m \leq k/2 \), the last assertion will follow.

By Proposition 2.6, without loss of generality, we may assume that \( \equiv H \) is the equality relation. By Lemma 3.3, in polynomial time, we can check that \( H \in \text{InAut}(C_p) \), and obtain a generator matrix \( M \) of \( \gamma(H) \) in standard form. If \( w \in W \) is in \( \text{InAut}(\gamma(H)) \), then \( M' := Mw \) satisfies \( \langle M \rangle = \langle M' \rangle \). Since \( M \) is in standard form, the elements of \( \langle M \rangle \) are uniquely determined by their first \( s \) coordinates. So to test whether a given \( w \in W \) is in \( \text{InAut}(\gamma(H)) \), it suffices to describe which columns of \( M \) are mapped onto the first \( s \) columns of \( M' \) and how they are scaled, then the rest of the action of \( w \) is determined.

We find all such \( w = \text{diag}(d_1, d_2, \ldots, d_k) \rho(\kappa) \) in \( \text{InAut}(\gamma(H)) \) by considering all choices of \( J = \{1^{s-1}, 2^{s-1}, \ldots, s^{s-1}\} \) and \( v := (d_i)_{i \in J} \in (\mathbb{F}_p^*)^s \).

For each such choice of \( J \) and \( v \), let \( M' \) be the partially defined matrix with columns \( M'_{s,i} = d_{i-1}^{-1}M_{s,i}^{-1} \), for \( 1 \leq i \leq s \). Then each partial row of \( M' \) extends to a unique vector in \( \langle M \rangle \), which must be the corresponding row of \( M' \). If \( w \in \text{InAut}(\gamma(H)) \) then \( (Mw)_{s,j} \) is a scalar multiple of a column of \( M \). Therefore we test whether \( J \) and \( v \) yield an element of \( \text{InAut}(\gamma(H)) \) by testing, for all \( j > s \), if there exists a \( j^{s-1} \notin \{1^{s-1}, 2^{s-1}, \ldots, s^{s-1}\} \) and \( j^{s-1} \) such that \( M'_{s,j} = d_{j^{-1}}^{-1}M_{s,j^{-1}}^{-1} \).

If for some \( j \) there are no such \( M_{s,j}^{s-1} \) and \( d_{j^{-1}}^{-1} \), then this choice of \( J \) and \( v \) does not extend to an element of \( \text{InAut}(\gamma(H)) \), and we move on to the next. Note that since the \( H \)-orbits are pairwise inequivalent, for each \( j \), there is at most one possible choice for \( j^{s-1} \). If we have succeeded in choosing \( j^{s-1} \) and \( d_{j^{-1}}^{-1} \) for all \( j \), then

\[ (M_{s,j}^{-1} \text{ diag}(d_1, d_2, \ldots, d_k) \rho(\kappa))_{s,j} = d_{j^{-1}}^{-1}M_{s,j}^{s-1} = M'_{s,j} \quad \text{for } 1 \leq j \leq k, \]

and so \( w = \text{diag}(d_1, d_2, \ldots, d_k) \rho(\kappa) \in \text{InAut}(\gamma(H)) \). There are \( k(p - 1) \leq n \) choices of \( d_{j^{-1}}^{-1}M_{s,j}^{s-1} \) for each \( j \in \{s+1, \ldots, k\} \), so for each \( J \) and \( r \), this step can be done in polynomial time.

Let \( Y \) be the set of all elements \( w \in \text{InAut}(\gamma(H)) \) found as above. We claim that \( N_{S_n}(H) = \langle C_{S_n}(H), \Xi^{-1}(Y) \rangle \), where \( \Xi^{-1}(Y) \) is any preimage of \( Y \). To show that \( N_{S_n}(H) \leq \langle C_{S_n}(H), \Xi^{-1}(Y) \rangle \), let \( \nu \in N_{S_n}(H) \). Then there exists \( y \in Y \) such that \( M\Xi(\nu) = My \), so \( \Xi(\nu)y^{-1} \) stabilises \( M \) and hence fixes each \( v \in \gamma(H) \). So by
Lemma 3.5.2, \( \Xi^{-1}(\Xi(\nu)y^{-1}) = \nu\Xi^{-1}(y^{-1}) \in C_{S_n}(H) \), therefore \( \nu \in \langle C_{S_n}(H), \Xi^{-1}(Y) \rangle \). The converse containment is clear.

For the complexity claim, since there are \( O(k^s(p - 1)^s) \in O(n^s) \) choices for \( J \) and \( v \), the set \( Y \) can be computed in time \( 2^{O(s \log n + \log n)} \). Now by Lemma 3.5.4, the set \( \Xi^{-1}(Y) \) can be computed in \( 2^{O((s+1)\log n)} \) time, and by Lemma 2.1.2, \( C_{S_n}(H) \) can be computed in time \( 2^{O(\log n)} \). Therefore \( N_{S_n}(H) \) can be computed in time \( 2^{O((s+1)\log n)} \). \( \square \)

While searching for elements of \( \text{MAut}(\gamma(H)) \) in \( W \), the above result limits the depth of the search tree. In practice, we search for \( \text{MAut}(\gamma(H)) \) in \( P \cong S_k \), as we shall see in Section 4.3.

4.2. Normalising elements that act non-trivially on \( \mathcal{O} \)

Let \( L = B \rtimes K \) be as in Lemma 2.8 and recall that \( N_{S_n}(H) \leq L \). In Proposition 4.6, we show that given a \( \kappa \in K \), we can decide in polynomial time if there exists an element of \( N_{S_n}(H) \) which induces the same permutation as \( \kappa \) on the set \( \mathcal{O} \) of \( H \)-orbits. Recall that \( D \) is the group of all diagonal matrices in \( \text{GL}_k(p) \).

Lemma 4.3. Let \( F := \text{GL}_s(p) \times D \). Define an action of \( F \) on \( M(s, k, p) \) by

\[
M^{(R, d)} = R^{-1}Md \quad \text{for all } R \in \text{GL}_s(p) \text{ and } d \in D.
\]

Let \( \kappa \in K \) and let \( M, M' \in M(s, k, p) \) be generator matrices of \( \gamma(H) \) and \( \gamma(H^\kappa^{-1}) \) respectively. Then there exists a \( b \in B \) such that \( b\kappa \in N_{S_n}(H) \) if and only if there exists an \( f \in F \) such that \( M^f = M' \).

**Proof.** Such a \( b \) exists if and only if \( H \) and \( H^\kappa^{-1} \) are in the same \( B \)-orbit, or equivalently, by Lemma 3.5, \( \gamma(H) \) and \( \gamma(H^\kappa^{-1}) \) are in the same \( D \)-orbit. The result now follows from the fact that \( \langle M \rangle = \langle M' \rangle \) if and only if \( M \) and \( M' \) are in the same orbit under left multiplication by \( \text{GL}_s(p) \). \( \square \)

We shall decide if \( M' \in M^F \) by computing certain \( F \)-orbit representatives, which we now define. For \( A, A' \in M(s, k, p) \), let \( A_{s, i}^R \) denote \( A_{s, i} \) reversed. We define \( A \prec A' \) if there exists a \( j \) such that \( A_{s, i} = A'_{s, i} \) for \( 1 \leq i < j \) and \( A_{s, j}^{R} \prec_{\text{lex}} A'_{s, j}^{R} \). We choose the representative of the orbit \( A^F \) to be the least element under the ordering \( \prec \).

Feulner gives an algorithm to compute such \( F \)-orbit representatives [6, Algorithm 1]. Since we will be proving its complexity, we will present our minor variation on the algorithm here. A key ingredient is the support partition, which we define next. We denote the tuple \((1, 2, \ldots, i)\) by \( \vec{i} \). The support of \( v \in \mathbb{F}_p^a \) is

\[
\text{Supp}(v) := \{i \mid 1 \leq i \leq a, v_i \neq 0\},
\]

and the support of \( V \leq \mathbb{F}_p^{a_0} \) is \( \text{Supp}(V) = \cup_{v \in V} \text{Supp}(v) \).
Definition 4.4. Let $M \in M(s,k,p)$ be in standard form. For $1 \leq j \leq k$, the support partition $Q_j$ is the finest partition of \{1, 2, \ldots, s\} such that for $1 \leq i \leq j$, there exists a cell $Q$ of $Q_j$ that contains $\text{Supp}(M_{s,i})$.

We present a simplified version of [6, Algorithm 1] as Algorithm 1. For $1 \leq j \leq k$, let $D^{[j]}$ be the subgroup of $D$ consisting of matrices of the form $\text{diag}(d_1, d_2, \ldots, d_j, 1, \ldots, 1)$, where $d_i \in F_p^*$. Algorithm 1 iteratively computes the orbit representative under the group $\text{GL}_j(p) \times D^{[j]}$, for increasing values of $j$.

**Algorithm 1 Computing the $F$-orbit representative of $A$.**

**Input:** $A \in M(s,k,p)$ in standard form.

**Output:** $F$-orbit representative of $A$.

1: Compute the partitions $Q_i$ for $s \leq j \leq k$ from Definition 4.4
2: for $j \in [s, s+1, \ldots, k-1]$ and $i \in [s, s-1, \ldots, 1]$ do $\triangleright$ see proof of Theorem 4.5
3: $Q, a \leftarrow [i]Q_j, A_{i,j}$
4: if $a \neq 0$ and $A_{r,j+1} \neq 0$ for all $r \in Q$ then
5: for $r \in Q$ do Multiply row $A_{r,*}$ by $a^{-1}$ end for $\triangleright O(sk)$ multiplications
6: for all $l \in [1, 2, \ldots, j]$ do $\triangleright O(k)$ time
7: if $\exists r \in Q$ s.t. $A_{r,l} \neq 0$ then Multiply column $A_{*,l}$ by $a$ end if $\triangleright O(s)$ time
8: end for
9: end if
10: end for
11: return $A$

**Theorem 4.5.** Let $A$ be a matrix in $M(s,k,p)$ in standard form. Then, assuming constant time field operations, Algorithm 1 returns the representative of $A^F$ in time polynomial in $k$.

**Proof.** Algorithm 1 and [6, Algorithm 1] are identical, except: Feulner considers codes over arbitrary finite fields, but we restrict to fields of prime order; Feulner’s algorithm performs row reduction, computes the partitions $Q_j$ and computes the $F$-orbit representative simultaneously, but we assume $A$ is in standard form, and first compute all the $Q_j$. Feulner’s algorithm also computes the orbit representative under $\text{GL}_j(p) \times D^{[j]}$ for $j < s$. We observe that since $A$ is in standard form, $A_{s,*}$ is an identity matrix, so $A$ is the orbit representative of $A$ under the action of $\text{GL}_s(p) \times D^{[s]}$. Hence the correctness of Algorithm 1 follows from the analysis in [6].

For the complexity claim, since $s \leq k$ and the analysis of Lines 2–10 of Algorithm 1 is straightforward, it remains only to show that the $Q_j$ can be computed in time polynomial in $sk \in O(k^2)$. As $A_{s,*}$ is an identity matrix, $Q_s$ is the trivial partition of \{1, 2, \ldots, s\} with $s$ cells. Now let $j \geq s+1$ and suppose that we have constructed $Q_{j-1}$. To construct $Q_j$, we merge all cells of $Q_{j-1}$ that have non-trivial intersection with $\text{Supp}(A_{s,j})$. Since $Q_{j-1}$ has at most $s$ cells, we can do this in time polynomial in $sk$. °

Lastly, we see how we use Algorithm 1 to decide if there exists an element of $N_{S_n}(H)$ which induces a given permutation of the $H$-orbits.
Proposition 4.6. Let $H$ be a subgroup of $S_n$ in $\mathfrak{An}(C_p)$, let $L = B \rtimes K$ be as in Lemma 2.8, and let $\kappa \in K$. Then in time polynomial in $n$, we can determine if there exists $b \in B$ such that $b\kappa \in N_{S_n}(H)$, and if so, output $b$.

Proof. By Lemma 3.3, in polynomial time, we can verify that $H \in \mathfrak{An}(C_p)$ and construct generator matrices $M, M' \in M(s, k, p)$ for $\gamma(H)$ and $\gamma(H^{-1})$ respectively, in standard form. By Lemma 4.3, such a $b$ exists if and only if $M$ and $M'$ have the same $F$-orbit representative. So by Theorem 4.5, in time polynomial in $n/p$, we can decide if such a $b$ exists. Furthermore, by keeping track of the changes in Lines 5 and 7 of Algorithm 1, we can simultaneously obtain elements $(R_1, d_1)$ and $(R_2, d_2)$ of $F$ which map $M$ and $M'$ to their $F$-orbit representatives. Let $\zeta : B \to D$ be as in Lemma 3.5. Then $HK^{-1}(d_1d_2^{-1}) = H^{-1}$ and hence we can construct $b = \zeta^{-1}(d_1d_2^{-1})$, in time polynomial in $n$ by Lemma 3.5.4. \(\square\)

4.3. Computing $N_{S_n}(H)$ by searching in $K$

In this subsection, we will first show that we can efficiently compute $N_B(H)$, then prove that $N_{S_n}(H)$ can be computed in time $O((\frac{n}{p})! + n^c)$, for some constant $c$.

Definition 4.7. Let $M \in M(s, k, p)$ be the generator matrix of $\gamma(H)$ in standard form. For $1 \leq i \leq s$, we call $x_i := \gamma^{-1}(M_i,s)$ the standard generators of $H$. We replace the original generating set of $H$ by the standard generating set $X = \{x_1, x_2, \ldots, x_s\}$.

We now show that each element of $N_B(H)$ conjugates elements of $X$ with non-disjoint supports to the same power. Recall that we identify $\mathbb{F}_p$ with the integers $\{0, 1, \ldots, p-1\}$.

Lemma 4.8. Let $b \in N_B(H)$, and let $x$ and $y$ be standard generators for $H$ with non-disjoint supports. Then there exists an $a \in \mathbb{F}_p^*$ such that $x^b = x^a$ and $y^b = y^a$.

Proof. As $x^b \in H$, the image $\gamma(x^b)$ is a linear combination of the rows of $M$. Since $M_{s, \pi}$ is an identity matrix and each element of $\langle M \rangle$ is completely defined by its first $s$ coordinates, there exists an $a \in \mathbb{F}_p^*$ such that $\gamma(x^b) = \gamma(x)^a$ and so $x^b = x^a$. Similarly, there exists an $a'$ such that $y^b = y^{a'}$. Now consider an $H$-orbit $\Omega_i$ contained in the intersection $\text{Supp}(x) \cap \text{Supp}(y)$ and let $g_i$ be the corresponding generator of the enveloping group $G$. Since $b$ fixes each $\Omega_i$ setwise, $g_i^b = g_i^a = g_i^{a'}$ and so $a = a'$. \(\square\)

For $Q \leq S_n$, a direct product decomposition $Q = R_1 \times R_2 \times \cdots \times R_r$ of $Q$ is a finest disjoint direct product decomposition of $Q$ if the groups $R_i$ have pairwise disjoint supports, and each $R_i$ cannot be written as a non-trivial direct product of subgroups with disjoint supports. We use such decompositions to compute the subgroup of $N_{S_n}(H)$ which setwise stabilises each $H$-orbit.
**Proposition 4.9.** Let $H$ be a subgroup of $S_n$ in class $\mathfrak{I}\mathfrak{P}(C_p)$ with orbits $\Omega_1, \Omega_2, \ldots, \Omega_k$, and let $B = \langle N_{\text{Sym}(\Omega_i)}(H|_{\Omega_i}) \mid 1 \leq i \leq k \rangle$. Then $N_B(H)$ can be computed in polynomial time.

**Proof.** Let $t$ be a primitive element of $\mathbb{F}_p^*$, and let $H = R_1 \times R_2 \times \cdots \times R_r$ be the finest disjoint direct product decomposition of $H$. For $1 \leq i \leq r$, let $\Gamma_i = \text{Supp}(R_i)$ and let $\sigma_i \in \text{Sym}(\Gamma_i)$ be such that $g_j^{\sigma_i} = g_j^t$ for all generators $g_j$ of $G$ such that $\Omega_j \subseteq \Gamma_i$. We first show that

$$N_B(H) = \langle G, \sigma_1, \sigma_2, \ldots, \sigma_r \rangle.$$  

$\geq$: Since $B$ fixes each $H$-orbit setwise,

$$C_B(H) \leq C_{\text{Sym}(\Omega_i)}(H|_{\Omega_i}) \times \cdots \times C_{\text{Sym}(\Omega_k)}(H|_{\Omega_k}) = G,$$

and so $G = C_B(H) \leq N_B(H)$.

Let $h \in H$. Then $h^{\sigma_1} = h_1^{\sigma_1} h_2^{\sigma_1} \cdots h_r^{\sigma_1}$ with $h_j \in R_j$. Fix $1 \leq j \leq r$. If $\Omega_i \subseteq \Gamma_j$ then $h_j|_{\Omega_i} = g_j^a$ for some $a$, and so

$$(h_j^{\sigma_j})|_{\Omega_i} = (h_j|_{\Omega_i})^{\sigma_j} = (g_j^a)^{\sigma_j} = g_j^{a^t} = (h_j|_{\Omega_i})^t.$$  

Since $t$ is independent of $\Omega_i$, the permutation $h_j^{\sigma_j} = h_j^t$ is in $R_j$. For $i \neq j$, since $h_i$ and $\sigma_j$ have disjoint supports, $h_i^{\sigma_j} = h_i$. Hence $h^{\sigma_j} \in H$ and so $\sigma_j \in N_B(H)$.

$\leq$: Let $b \in N_B(H)$ and let $X$ be the standard generating set of $H$. By Lemma 4.8, there exists $t' \in \mathbb{F}_p^*$ such that $x^{b|_{\Gamma_i}} = x^{t'}$ for all $x \in X$ whose support intersects $\Gamma_i$ non-trivially. So $(x|_{\Gamma_i})^{b|_{\Gamma_i}} = (x|_{\Gamma_i})^{t'} = (x|_{\Gamma_i})^{a^t}$. Since $R_i = \langle (x|_{\Gamma_i}) \mid x \in X \rangle$, the permutation $b|_{\Gamma_i}$ is in $\langle C_B(R_i), \sigma_i^t \rangle$, which is a subgroup of $\langle G, \sigma_i \rangle$, so the result follows.

For the complexity result, in time polynomial in $p$, we can determine a primitive $t \in \mathbb{F}_p^*$. In time polynomial in $n$, we can compute the unique finest disjoint direct decomposition of $H$, by [4], and construct the $g_i$, by Lemma 3.3. The $\text{Sym}(\Omega_i)$-conjugacy of permutations can be solved in polynomial time, so we can construct the $\sigma_i$ in polynomial time. \[ \square \]

Lastly, we show that to compute $N_{S_n}(H)$ for $H \in \mathfrak{I}\mathfrak{P}(C_p)$, it suffices to search in $K$. Our implementation, which will be described in Section 6, uses the procedure described in the following proof.

**Theorem 4.10.** Norm-Sym for $H = \langle X \rangle \leq S_n$, where $H \in \mathfrak{I}\mathfrak{P}(C_p)$, can be solved in time $2^{O\left(\frac{n \log n}{p} + \log n\right)}$.

**Proof.** By Lemma 3.3, in polynomial time $2^{O\left(\log n\right)}$, we can recognise that $H \in \mathfrak{I}\mathfrak{P}(C_p)$, and obtain a generating set $\langle \phi_i \mid 2 \leq j \leq k \rangle$ for $K$, an isomorphism $\gamma : G \to \mathbb{F}_p^k$ as in Definition 3.2, and a generator matrix $M$ for $\gamma(H)$ in standard form. Next we compute
\( C_{S_n}(H) \) and \( N_B(H) \), in time polynomial in \( n \) by Lemma 2.1.1 and Proposition 4.9 respectively.

We initialise \( N \) as \( \langle C_{S_n}(H), N_B(H) \rangle \). For each \( \kappa \in K \), we determine if there exists \( b \in B \) such that \( bk \in N_{S_n}(H) \), in time polynomial in \( n \) by Proposition 4.6. If such a \( b \) exists, we update \( N \) as \( \langle N, bk \rangle \). This takes time \( O((\frac{n}{b})!n^c) = 2^{O(\frac{n}{b} \log \frac{n}{b} + \log n)} \), for some constant \( c \).

By the end of the procedure, we have \( N \leq N_{S_n}(H) \). To show \( N_{S_n}(H) \leq N \), let \( \nu \in N_{S_n}(H) \). Then since \( \nu \in L = B \times K \) by Lemma 2.8.3, we can find \( \kappa \in K \) and \( b \in B \) such that \( \nu = bk \) by Lemma 2.8.4. If \( \kappa = 1 \), then \( \nu \in N_B(H) \) and so \( \nu \in N \). Suppose now that \( \kappa \neq 1 \). We have already found an element \( b' \in B \) such that \( b'\kappa \in N_{S_n}(H) \), so \( b'\kappa \in N \). Then \( bk(b'\kappa)^{-1} = bb^{-1} \in N_B(H) \leq N \). Therefore \( \nu = bk \in N \). 

Theorem 1.4 now follows. Note that Proposition 4.2 gives better complexity than Theorem 4.10 when \( p \leq k \). However, the algorithm for Proposition 4.2 requires the checking of all elements of \( (F_p^*)^m \), which we do not have useful pruning methods for. When \( p \) and \( m \) are large, this becomes infeasible (see Table 1).

5. Pruning techniques

Recall that we compute \( N_{S_n}(H) \) using backtrack search. In this section, we present some methods to compute efficiently using \( \gamma(H) \), including some pruning techniques. We will see how to apply these results in Section 6. Throughout, let \( H \) be as in Hypothesis 3.1.

First we show that stabilisers and the relation \( \equiv_H \) from Definition 2.3 can be computed using a generating matrix \( M \) for \( \gamma(H) \). Recall that we denote the rows and columns of \( M \) by \( M_{i,*} \) and \( M_{*,j} \), and write \( \vec{s} \) for \( \{1, 2, \ldots, s\} \).

**Lemma 5.1.** Let \( \beta \) be a point in an \( H \)-orbit \( \Omega_j \). If \( M_{*,j} \) is non-zero in a unique row, \( i \) say, then \( \gamma(H(\beta)) = \langle M_{\vec{s}\setminus i,*} \rangle \).

**Proof.** Since \( M_{\vec{s}\setminus i,j} \) is the zero vector, \( \gamma^{-1}(M_{\vec{s}\setminus i,*}) \) fixes \( \beta \), so \( \gamma(H(\beta)) \geq \langle M_{\vec{s}\setminus i,*} \rangle \). For the other containment, let \( h \in H(\beta) \). Since \( H|_{\Omega_j} \) is regular, \( (H|_{\Omega_j})(\beta) = 1 \), so \( \gamma(h)j = 0 \). Since \( M_{i,*} \) is the only row of \( M \) with non-zero entry in the \( j \)-th position, \( \gamma(h) \in \langle M_{\vec{s}\setminus i,*} \rangle \). 

The assumption on \( M_{*,j} \) in the previous lemma can be achieved by performing row operations. Hence any point stabiliser can be computed by row operations, which in practice are faster than the equivalent permutation group calculations.

**Lemma 5.2.** Let \( M \) be a generator matrix of \( \gamma(H) \), and let \( a_i \) and \( a_j \) be the first non-zero entries of \( M_{*,i} \) and \( M_{*,j} \) respectively. Then \( \Omega_i \equiv_H \Omega_j \) if and only if \( a_i^{-1}M_{*,i} = a_j^{-1}M_{*,j} \).

**Proof.** We show that \( \Omega_i \equiv_H \Omega_j \) if and only if \( M_{*,j} = aM_{*,i} \) for some \( a \in F_p^* \), from which the result follows.

\( \Leftarrow \): Recall the \( p \)-cycles \( g_i \) from Hypothesis 3.1. Fix \( \alpha \in \Omega_i \) and \( \beta \in \Omega_j \). Define a mapping
\[ \psi : \Omega_i \to \Omega_j \] by setting \( \psi(\alpha^{9^u}) = \beta^{9^u} \) for \( 0 \leq u \leq p - 1 \). Since \( \langle g_i \rangle \) and \( \langle g_j \rangle \) are isomorphic and regular, \( \psi \) is a bijection, and the reader may check that \( \psi \) witnesses the equivalence of \( \Omega_i \) and \( \Omega_j \).

By Definition 2.3, there exists an involution \( \sigma \in \text{Sym}(\Omega_i \cup \Omega_j) \) such that \( h|_{\Omega_j} = (h|_{\Omega_i})^\sigma \) for all \( h \in H \). Let \( a \) be such that \( g_i^a = g_j^a \), let \( h \in H \), and let \( v = \gamma(h)_i \) and \( u = \gamma(h)_j \). Then

\[ (g_i^a)^v = (g_i^a)^v = (h|_{\Omega_i})^\sigma = h|_{\Omega_j} = g_j^v, \quad \text{so} \ u = av. \]

Hence we can use elementary linear algebra to compute the \( \equiv_H \)-classes.

Next, we show how we can use linearly dependent columns of \( M \) to prune the search tree. For a subset \( J \) of \( \{1, 2, \ldots, k\} \), we denote the union \( \cup_{j \in J} \Omega_j \) by \( \Omega_J \). A set \( V \) of linearly dependent vectors is minimally linearly dependent if no proper subset \( U \subset V \) is linearly dependent.

**Lemma 5.3.** Let \( I \) and \( J \) be subsets of \( \{1, 2, \ldots, k\} \) such that there exists \( \nu \in N_{S_n}(H) \) with \( \Omega_I^\nu = \Omega_J \). Then the rank of \( M_{*,I} \) is equal to the rank of \( M_{*,J} \). Hence the columns of \( M_{*,I} \) are minimally linearly dependent if and only if the columns of \( M_{*,J} \) are minimally linearly dependent.

**Proof.** The assumption that \( \nu \) exists implies that \( H|_{\Omega_I} \) and \( H|_{\Omega_J} \) are conjugate in \( \text{Sym}(\Omega_I \cup \Omega_J) \). Let \( r_I \) and \( r_J \) be the ranks of \( M_{*,I} \) and \( M_{*,J} \) respectively. Then \( p^{r_I} = |(H|_{\Omega_J})| = |(H|_{\Omega_J})| = p^{r_J} \), so \( r_I = r_J \).

Observe that the columns of \( M_{*,I} \) are minimally linearly dependent if and only if \( M_{*,I} \) has rank \( |I| - 1 \) and each proper subset \( S \) of columns has rank \( |S| \). Since rank is preserved by conjugation, the final claim follows.

The following lemma is elementary but extremely useful for pruning the search tree \( T \). Two columns \( M_{*,I} \) and \( M_{*,J} \) of \( M \) are equivalent, denoted by \( M_{*,I} \equiv_M M_{*,J} \), if \( \Omega_i \equiv_H \Omega_j \). The weight enumerator \( w(C) \) of a linear code \( C \leq F_p^k \) is the polynomial

\[ w(C) = \sum_{i=1}^k w_i x^i, \]

where \( w_i \) is the number of codewords of weight \( i \).

**Lemma 5.4.**

1. Let \( \Delta \) and \( \Gamma \) be subsets of \( \Omega \) such that \( |H_(\Delta)| = |H_(\Gamma)| = p^l \), say. Denote generator matrices of \( H_(\Delta) \) and \( H_(\Gamma) \) by \( M_(\Delta), M_(\Gamma) \in M(l,k,p) \). If there exists \( \nu \in N_{S_n}(H) \) such that \( \Delta^\nu = \Gamma \), then there exists a bijection between the \( (\equiv_{M_(\Delta)}) \) -classes and the \( (\equiv_{M_(\Gamma)}) \) -classes that preserves the class sizes.
2. Let \( Q \) and \( Q' \) be \( S_n \) -conjugate subgroups of \( G \). Then \( w(\gamma(Q)) = w(\gamma(Q')) \).
3. Let \( V \) be the set of non-zero minimum weight vectors in \( \gamma(H) \). Then \( N_{S_n}(H) \) setwise stabilises \( \gamma^{-1}(V) \). Hence \( N_{S_n}(H) \) setwise stabilises the partition of \( \{1, 2, \ldots, k\} \) where \( i \sim j \) if and only if \( \{v \in V \mid v[i] \neq 0\} = \{v \in V \mid v[j] \neq 0\} \).
Proof. For Part 1, by Lemma 2.2.1, the existence of \( \nu \) implies that \( H(\Delta) \) and \( H(\Gamma) \) are conjugate in \( S_n \), so the required bijection exists by Lemma 2.5. For Parts 2 and 3, notice that \( \gamma(h)_i \neq 0 \) if and only if \( h|_{\Omega_i} \neq 1 \). So the weight of \( \gamma(h) \) is the number of \( p \)-cycles of \( h \). \( \Box \)

There is no known polynomial-time algorithm for computing the weight enumerator, so we use a simple heuristic to determine when we use Part 2 (see Algorithm 5).

Lastly, we give a technical lemma that can be used to prune the search tree \( T \), whose rationale is as follows. Assume we are at a node \( \tau \) at depth \( m > s \) in \( T \). If there exists a \( \kappa \in K \) represented by a leaf below \( \tau \) such that \( b\kappa \in N_{S_n}(H) \), then there exists \( d \in D \) such that \( d\kappa \in MAut(\gamma(H)) \). Let \( M' := M d\kappa \). Then since \( m > s \), at node \( \tau \) we know, up to \( s \) unknown scalars from \( F_p^* \), the first \( s \) columns of \( M' \). Since rows of \( M' \) are linear combinations of the rows of \( M \) and the elements of \( \langle M \rangle \) are defined by their first \( s \) coordinates, we now know the whole of \( M' \), up to \( s \) unknown scalars. So if we can deduce that some entries of \( M' \) must be zero, then we can sometimes show that no such \( \kappa \) exists, and hence we can backtrack from \( \tau \).

Lemma 5.5. Let \( M \in M(s,k,p) \) be a generator matrix of \( \gamma(H) \) in standard form. Fix \( m \in \{s,s+1,\ldots,k-1\} \) and let \( J = \{1,2,\ldots,m\} \cup \{u\} \subseteq \{1,2,\ldots,k\} \). Let \( f \) be an injection from \( J \) to \( \{1,2,\ldots,k\} \). If there exists an \( i \in \{1,2,\ldots,s\} \) such that

\[
M_{i,f(u)} \neq 0 \text{ and } M_{i,f(j)} M_{j,u} = 0 \quad \text{for all } j \in J \cap \{1,2,\ldots,s\},
\]

(3)

then there does not exist \( \nu \in N_{S_n}(H) \) such that \( \Omega_j^\nu = \Omega_{f(j)}^\nu \) for all \( j \in J \).

Proof. Aiming for a contradiction, assume that such an \( i \) and \( \nu \) exist. Then by Lemma 3.5.3, \( \Xi(\nu) \in MAut(\gamma(H)) \). So there exist \( d = \text{diag}(d_1,d_2,\ldots,d_k) \in D \) and \( q \in P \) such that \( \Xi(\nu) = dq \). Let \( M' = Mdq \). Then by considering the action of \( W \) on \( F_p^k \),

\[
M'_{i,j} = d_{f(j)} M_{i,f(j)}, \quad \text{for all } j \in J \text{ and all } i.
\]

(4)

In particular, \( M'_{i,u} = d_{f(u)} M_{i,f(u)} \), which is non-zero.

Each row of \( M' \) is a linear combination of the rows of \( M \), and \( M \) is in standard form, so

\[
M'_{i,*} = \sum_{l=1}^{s} M'_{i,l} M_{l,*} = \sum_{l=1}^{s} d_{f(l)} M_{i,f(l)} M_{l,*},
\]

where the second equality follows from (4). Therefore by (3), \( M'_{i,u} = \sum_{l=1}^{s} d_{f(l)} M_{i,f(l)} \times M_{l,u} = 0 \), a contradiction. \( \Box \)
6. Description of implemented algorithm

Given a generating set $X$ of $H \leq S_n$ such that $H \in \mathfrak{Inn}(C_p)$, we compute $N_{S_n}(H)$ using Algorithm 2. The algorithm roughly follows the description in the proof of Theorem 4.10, with extra steps for pruning the search tree.

We first represent $H$ and $H^\perp$ by generator matrices $M$ and $M^\perp$ respectively, where $H^\perp$ is as in Section 4.1. Then we compute $C_{S_n}(H)$, which may allow us to reduce to a group with a smaller degree, as in Proposition 2.6. Next we compute $N_B(H)$, where $B$ is as in Lemma 2.8: that is, we compute all normalising elements fixing every $H$-orbit setwise. Lastly, we perform backtrack search in $K \cong S_k$ to find the remaining normalising elements.

To simplify notation, we assume in the description below that $\gamma(H)$ has dimension $s \leq k/2$ and that $H$ has no equivalent orbits.

**Algorithm 2** Computing the normaliser of $H \in \mathfrak{Inn}(C_p)$.

**Input:** A generating set $X$ of $H \leq S_n$, where $H$ has no equivalent orbits.

**Output:** $N_{S_n}(H)$

1: if $H \not\in \mathfrak{Inn}(C_p)$ then return FAIL end if \hspace{1cm} \triangleright \text{polynomial by Lemma 3.3}
2: Compute the enveloping group $G$ of $H$ and $\gamma : G \to \mathbb{F}_p$ \hspace{1cm} \triangleright \text{polynomial by Lemma 3.3}
3: Let $\mathcal{O} = \{\Omega_1, \Omega_2, \ldots, \Omega_t\}$ be the orbits of $H$, ordered as in Hypothesis 3.1
4: Let $M$ be a generator matrix of $\gamma(H)$ in standard form
5: Let $M^\perp$ be a generator matrix of $\gamma(H^\perp)$
6: $N \leftarrow N_B(H)$ \hspace{1cm} \triangleright \text{using Proposition 4.9}
7: $\text{doms} \leftarrow \text{DOMAINSINIT}(M, M^\perp)$ \hspace{1cm} \triangleright \text{see Algorithm 3}
8: $\text{LDScols} \leftarrow \{\{i\} \cup \{j \mid M_{j,i} \neq 0, 1 \leq j \leq s\} \mid s + 1 \leq i \leq k\}$
9: $\text{dualLDScols} \leftarrow \{\{i\} \cup \{j \mid (M^\perp)_{j,i} \neq 0, k - s + 1 \leq j \leq k\} \mid 1 \leq i \leq k - s\}$
10: $\text{RECURSEARCH}([], \text{doms})$ \hspace{1cm} \triangleright \text{see Algorithm 4}
11: return $N$

**Lines 1–9 of Algorithm 2: Pre-search**

Before the backtrack search, we compute various structures we shall use later.

**Lines 4–5** We compute a row reduced generator matrix $M$ of $\gamma(H)$ using $X$. Since $M$ is in standard form, $M = (I_s \mid M_0)$, and $M^\perp := (M_0^T \mid I_{k-s})$ is a generator matrix of $\gamma(H)^\perp$ [19].

**Line 6** We gather the normalising elements we find in a group $N$. Since we assume that $H$ has no equivalent orbits, $G = C_{S_n}(H) \leq N_B(H)$ by Lemma 2.4.

**Line 7** For each $H$-orbit $\Omega_i$, we compute a set of possible images of $\Omega_i$ under $N_{S_n}(H)$, called the domain of $\Omega_i$. This will be used to guide our search in Algorithm 4.

**Lines 8–9** We compute certain sets of linearly dependent columns of $M$ and $M^\perp$: we will later explain how we use these sets for pruning. Since $M$ is in standard form, the standard basis of $\mathbb{F}_p^s$ (as column vectors) forms the first $s$ columns. Therefore we can write each later column of $M$ as a linear combination of $M_{s,1}, M_{s,2}, \ldots, M_{s,s}$. Similarly, the standard basis of $\mathbb{F}_p^{k-s}$ forms the last $k - s$ columns of $M^\perp$. 
By the end of Line 6, $N$ contains $N_B(H) \geq C_{S_n}(H)$. So, as in Theorem 4.10, this allows us to only search for non-trivial elements of $K$ which give rise to elements of $N_{S_n}(H)$.

Algorithm 3: Initialising the domains with domainsInit, makeDualEquivPartn, makeStabsPartn and makeInvSetPartn

DOMAINSInit returns a list of sets doms, where each entry doms[i] is a subset of \{1, 2, \ldots , k\} such that if there exists $\nu \in N_{S_n}(H)$ with $\Omega_j = \Omega_i$ then $j \in \text{doms}[i]$. We shall compute partitions $P_v^\perp$, $P_s$, $P_s^\perp$, $P_I$ of \{1, 2, \ldots , k\}, each representing a partition of $O = \{\Omega_1, \Omega_2, \ldots , \Omega_k\}$ preserved by $N_{S_n}(H)$. Hence their meet $P_*^*$ is also preserved by $N_{S_n}(H)$ and we initialise the domain of $i$ to be $[i]P_*^*$.

We have assumed that the $H$-orbits are pairwise inequivalent, but the relation $\equiv_H^\perp$ may be non-trivial. makeDualEquivPartn exploits this and computes $P_v^\perp$, which $N_{S_n}(H)$ preserves by Lemma 2.5. Since known subgroups of $N_{S_n}(H)$ can be used to prune the search tree, and centralising elements for $H^\perp$ that act non-trivially on the set of $H^\perp$-orbits yield elements of $N_{S_n}(H)\setminus B$, we also update $N$ with such normalising elements.

makeStabsPartn considers the stabiliser of each $H$- and $H^\perp$-orbit, then uses the sizes of these stabilisers’ $\equiv$-classes to make partitions $P_s$ and $P_s^\perp$, which are preserved by $N_{S_n}(H)$ by Lemmas 5.4.1 and 4.1.

makeInvSetPartn first computes the set invSet of minimal weight vectors of $\gamma(H)$. The algorithm systematically considers linear combinations of rows of $M$ with increasingly many non-zero coefficients. Since $M_{\gamma,\gamma}$ is the identity matrix, if $v$ is a linear combination with non-zero coefficients of $i$ rows of $M$, then $\text{wt}(v) \geq i$. As we are only interested in vectors with weight at most $m$, the current minimum weight of codewords found so far, we only consider all such linear combinations $v$ of up to $m$ rows of $M$. The group $N_{S_n}(H)$ preserves $P_I$ by Lemma 5.4.3.

Algorithm 4: Search with recurseSearch

Now we shall describe the recursive search, which is initialised in Line 10 of Algorithm 2. We shall traverse the search tree depth first, using the domains doms to guide our search. Note that $N$ is a global variable that stores the group generated by all elements of $N_{S_n}(H)$ we have found so far. The pruning tests used in Lines 10–16 are presented as Algorithm 5.

Lines 2–7 If $d = k$, then we have arrived at a leaf node of the search tree. We only require a generating set of $N_{S_n}(H)$, so we can backtrack to node $[\alpha_1, \alpha_2, \ldots , \alpha_{j-1}]$. See [18, §9.1.1] for details.

Line 9 By [18, §9.1.1], it suffices to test only the minimum value of each $N(\Omega_1, \Omega_2, \ldots , \Omega_{d-1})$-orbit.
Algorithm 3 Initialise domains.

1: procedure DOMAINSInit($M, M^\perp$)
2: \[ P^+, P^+, P^+_2, P_I \leftarrow \text{MAKEDUALEQUIVPartn}(M^\perp), \text{MAKESTABSPartn}(M), \text{MAKESTABSPartn}(M^\perp), \]
3: \[ P^* \leftarrow \text{Meet}(P^+, P^+, P^+_2, P_I) \]
4: return \[ [i]_{P^*} \mid 1 \leq i \leq k \]
5: end procedure

6: procedure MAKEDUALEQUIVPartn($M^\perp$)
7: Compute the \((\equiv_{H^\perp})\)-classes \( \triangleright \text{using Lemma 5.2} \)
8: for each pair of equivalent orbits \( \Omega_i \) and \( \Omega_j \) do
9: Let \( c \in C_{\Omega_i}(H^\perp) \) conjugate \( \Omega_i \) to \( \Omega_j \) \( \triangleright \text{as in Lemma 2.4} \)
10: Find \( b \in B \) and \( \kappa \in K \) such that \( c = b\kappa \) \( \triangleright \text{using Lemma 2.8.4} \)
11: \[ N \leftarrow \langle N, b^{-1}\kappa \rangle \]
12: \( \triangleright b^{-1}\kappa \in N_{\Omega_i}(H) \) by Lemma 4.1
13: end for
14: return partition of \( \{1, 2, \ldots, k\} \) such that \( i \sim j \) if and only if \( ||\Omega_i||_\equiv_{\mu^+} = ||\Omega_j||_\equiv_{\mu^+} || \)
15: end procedure

16: procedure MAKESTABSPartn($M$) \( \triangleright \text{will apply to both } M \) and \( M^\perp \)
17: for \( i \in \{1, 2, \ldots, k\} \) do
18: \[ Q_i \leftarrow \gamma^{-1}((M))_{\Omega_i} \]
19: \( \triangleright \text{using Lemma 5.1} \)
20: \[ Q_i \leftarrow \text{the multiset of sizes of the } (\equiv_{Q_i})\text{-classes} \]
21: \( \triangleright \text{using Lemma 5.2} \)
22: end procedure

23: procedure MAKEINVSetPartn($M$)
24: \[ m \leftarrow \min_{1 \leq i \leq s}(\wt(M_{i, \ast})) \]
25: \( \triangleright \text{minimum weight of codewords found so far} \)
26: \[ \text{invSet} \leftarrow \{ \text{invSet} \mid \wt(M_{i, \ast}) = m \} \]
27: \( \triangleright \text{minimum weight codewords found so far} \)
28: for \( i \in \{2, 3, \ldots, m\} \) do
29: \( \triangleright \text{for all linear combinations } v \text{ with non-zero coefficients of } i \text{ rows of } M \) do
30: \[ \text{if } \wt(v) < m \text{ then } \text{Reset } m \leftarrow \wt(v) \text{ and } \text{invSet} \leftarrow \{v\} \text{ end if} \]
31: \[ \text{if } \wt(v) = m \text{ then } \text{Add } v \text{ to invSet} \text{ end if} \]
32: end procedure

Line 23 Since the images of elements of \( \mathcal{O} \) must be pairwise distinct, whenever the domains change, we can further refine them. For each \( i \), let

\[ J_i := \{ j \mid 1 \leq j \leq k \text{ and } \text{doms}[j] = \text{doms}[i] \} \]

If \( |J_i| = |\text{doms}[i]| \), then any normalising element under the current node maps the set \( \{\Omega_i \mid i \in J_i\} \) to \( \{\Omega_i \mid i \in \text{doms}[i]\} \). We remove elements of \( \text{doms}[i] \) from \( \text{doms}[t] \) for all \( t \notin J_i \).

Lines 24–25 If any domain becomes empty, we backtrack. Otherwise, we continue the depth-first search, branching using doms.

Algorithm 5: Pruning functions checkLDS and compareStabs

CHECKLDS uses minimally linearly dependent columns of \( M \) to prune the search tree. If the condition in Line 6 is satisfied then by Lemma 5.3, there is no normalising element under the current node that sends \( \Omega_I[1] \) to \( \Omega_i \), so we remove \( i \) from the domain of \( I[1] \).
Algorithm 4 RECURSESEARCH.

1: procedure RECURSESEARCH(α = [α₁, α₂, ..., αₙ], doms)
2:   if d = k then
3:     k ← permutation in K such that Ωₖ = Ωₖᵢ for all i
4:     if there exists b ∈ B such that bk ∈ Nₛₖ(H) then
5:       N ← {N, bk}
6:     Backtrack to [α₁, ..., αₙ], where j is the largest integer such that αᵢ = i for all i ≤ j
7:   end if
8:   else
9:     if [α₁, α₂, ..., αₙ₋₁] = [1, 2, ..., d − 1] and αₙ ≠ d and αₙ is not minimal in αₙ(Nα₁,α₂,...,αₙ₋₁) then
10:        return end if
11:     passed1, doms ← CHECKLDS(M, LDScols, α, doms) see Algorithm 5
12:     passed2, doms ← CHECKLDS(M⁺, dualLDScols, α, doms)
13:     Mstab, MstabIm ← γ(H(Ω₁,...,Ωₙ)), γ(H(Ω₁,...,Ωₙ))
14:     passed3, doms ← COMPARESTABS(Mstab, MstabIm, α, doms)
15:     Mstab⁺ ← generator matrix of γ((H⁺)(Ω₁,...,Ωₙ))
16:     passed4, doms ← COMPARESTABS(Mstab⁺, MstabIm⁺, α, doms)
17:     if ¬(passed1 and passed2 and passed3 and passed4) then return end if
18:     if d > s then
19:       for 1 ≤ i ≤ s and t ∈ [s + 1 ≤ t ≤ k] | Mᵢ,αₙ.Mᵢ,t = 0 for all u] do
20:         doms[t] ← [j | doms[t] | Mᵢ,j = 0] using Lemma 5.5
21:       end for
22:     end if
23:     if \( \exists i \) such that doms[i] = \emptyset then return end if
24:     for αₙ₊₁ ∈ doms[d + 1] do RECURSESEARCH([α₁, α₂, ..., αₙ₊₁], doms) end for
25:     end if
26:     end if
27: end procedure

COMPARESTABS uses conjugacy of point stabilisers to prune the search tree, as in Lemma 2.2.1. If any of the conditions in Lines 13, 15 or 18 is satisfied then there are no normalising elements under the current node, by Lemmas 2.5, 5.4.1 and 5.4.2 respectively.

Algorithm 5 CHECKLDS and COMPARESTABS.

1: procedure CHECKLDS(M, LDScols, [α₁, α₂, ..., αₙ], doms)
2:   for lds ∈ LDScols do see Algorithm 2, Line 8
3:     I ← lds \ [1, 2, ..., d] unassigned column images
4:     if |I| = |lds| − 1 then image of \( Mᵢ \) must be in the span of other columns
5:       for i ∈ doms[I[1]] do
6:         if \( Mᵢ,j \notin \{ Mᵢ,αₙ \mid j \in lds \setminus I[1] \} \) Remove i from doms[I[1]] end if
7:       end for
8:     end if
9:   end for
10:   return TRUE, doms
11: end procedure

12: procedure COMPARESTABS(Mstab, MstabIm, [α₁, α₂, ..., αₙ], doms)
13:   if the multisets of sizes of the (≡Mstab)-classes and the (≡MstabIm)-classes are different then
14:     FALSE, doms end if
15:   for i ∈ \{1, 2, ..., k\} \ [α₁, α₂, ..., αₙ] and j ∈ doms[i] do
16:     if \(|Ωᵢ|_{\equiv Mstab} \neq |Ωᵢ|_{\equiv MstabIm} \) then Remove j from doms[i] end if
17:   end for
18:   if (s − d) * p ≤ 45 then 45 is a heuristic
19:     if w(Mstab) \neq w(MstabIm) then return FALSE, doms end if
20:   end if
21:   return TRUE, doms no obstruction to conjugacy found
22: end procedure
7. Extension: groups in class $\mathfrak{G}(D_{2p})$

In this section, we consider $H \in \mathfrak{G}(D_{2p})$, where $p$ is an odd prime and $D_{2p}$ is the dihedral group of order $2p$ and degree $p$. We show that $N_{S_n}(H)$ can be found by computing the normalisers of its Sylow subgroups, which can be identified with groups in classes $\mathfrak{G}(C_p)$ and $\mathfrak{G}(C_2)$.

We will assume the following throughout this section. Let $n = pk$ and $\Omega = \{1, 2, \ldots, n\}$. Let $H$ be a subgroup of $S_n$ in class $\mathfrak{G}(D_{2p})$ with orbits $\Omega_1, \Omega_2, \ldots, \Omega_k$ such that $\Omega = \bigcup_{i=1}^k \Omega_i$. Let $D_i := H|\Omega_i$ for each $1 \leq i \leq k$, so $D = D_1 \times D_2 \times \cdots \times D_k \leq S_n$ is the enveloping group of $H$. For $1 \leq i \leq k$, let $G_i$ be the Sylow $p$-subgroup of $D_i$, let $G = G_1 \times G_2 \times \cdots \times G_k \leq S_n$, and let $H_q$ be a Sylow $q$-subgroup of $H$, for $q \in \{2, p\}$.

Lemma 7.1.

1. $H = H_p \rtimes H_2 = (H \cap G) \rtimes H_2$, and so $N_{S_n}(H) \leq N_{S_n}(H_p)$.
2. $H_p$ is a subdirect product of $G$, and so $H_p \in \mathfrak{G}(C_p)$.
3. For all $i$, there exists $\alpha_i \in \Omega_i$ such that $H_2|\Omega_i = (D_i)_{(\alpha_i)}$.

Proof. Part 1: As $G$ is the unique Sylow $p$-subgroup of $D$, it follows that $H_p \leq H \cap G$, so $H_p = H \cap G$. Since $G \unlhd D$, the group $H_p$ is characteristic in $H$, so the last assertion follows.

Part 2: Since $H$ is a subdirect product of $D$, for all $i$ there exists $h \in H$ such that $1 \neq h|\Omega_i \in G_i$. Then, for all $j$, the restriction $h^2|\Omega_j$ is of order 1 or $p$. So $h^2|\Omega_j \in G_j$ for all $j$ and therefore $h^2 \in G \cap H = H_p$. Since $h|\Omega_i$ is a $p$-cycle, $h^2|\Omega_i \neq 1$, so $H_p|\Omega_i = G_i$.

Part 3: For all $i$, there exists an involution $r_i \in D_i$ such that $H_2|\Omega_i \leq \langle r_i \rangle$. Since every involution in $D_i$ fixes a point, there exists $\alpha_i \in \Omega_i$ such that $r_i$ fixes $\alpha_i$. So $H_2|\Omega_i$ is a subgroup of $(D_i)_{(\alpha_i)}$. If $H_2$ pointwise stabilises $\Omega_i$, then by Part 1,

$$H|\Omega_i = (H_p H_2)|\Omega_i = H_p|\Omega_i = G_i,$$

but $H$ is a subdirect product of $D$, a contradiction, and so $H_2|\Omega_i = (D_i)_{(\alpha_i)}$. \square

For $2 \leq i \leq k$, let $\phi_i : \Omega_1 \to \Omega_i$ witness the permutation isomorphism from $D_1$ to $D_i$, and satisfy $\phi_i(\alpha_1) = \alpha_i$, where $\alpha_i$ is as in Lemma 7.1.3. Let $K = \langle \phi_i \mid 2 \leq i \leq k \rangle$, let

$$L = \langle N_{\text{Sym}(\Omega_1)}(G_1), \ldots, N_{\text{Sym}(\Omega_k)}(G_k), K \rangle \cong (C_p \rtimes C_{p-1}) \wr S_k$$

and let $I = N_L(H_p) \cap N_L(H_2)$.

Proposition 7.2. $N_{S_n}(H) = IH$. 
Proof. By Lemma 7.1.1, for all $h \in H$, there exist $h_p \in H_p$ and $h_2 \in H_2$ such that $h = h_ph_2^\prime$. So $h^\prime = h_2^\prime h_2 \in H_pH_2 = H$ for all $i \in I$. Therefore $IH \leq N_{S_n}(H)$.

For the converse containment, $N_{S_n}(H) = N_L(H)$ by Lemmas 7.1.1 and 2.8.3. Using the Frattini argument, $N_{S_n}(H) = N_{N_L(H)}(H_2)H = (N_L(H) \cap N_L(H_2))H$. Finally by Lemma 7.1.1, $N_L(H)$ is contained in $N_L(H_p)$, so $N_{S_n}(H) \leq IH$. □

For $1 \leq i \leq k$, let $N_i = N_{\text{Sym}(\Omega_i)}(H_2|\Omega_i) \cap N_{\text{Sym}(\Omega_i)}(G_i)$, and let $T$ be the group $\langle N_1, N_2, \ldots, N_k, K \rangle$.

Lemma 7.3.

1. $I \leq T$ and so $I = N_T(H_2) \cap N_T(H_p)$.
2. Let $t \in \mathbb{F}_p^\ast$ be primitive. For $1 \leq i \leq k$, let $g_i \in \text{Sym}(\Omega_i)$ be a generator of $G_i$, and let $c_i$ be an element of $\text{Sym}(\Omega_i\setminus\{\alpha_i\})$ such that $g_i^{c_i} = g_i^t$. Then $N_i = \langle c_i \rangle$ and so $T \cong C_{p-1} \rtimes S_k$.

Proof. Part 1: Let $i \in I$. Since $L \cong N_{\text{Sym}(\Omega_i)}(G_i) \cap S_k$ by Lemma 2.8.2, there exists $\kappa \in K$ such that $\kappa^{-1}$ fixes each $\Omega_i$ setwise and $(\kappa^{-1})|\Omega_i \in N_{\text{Sym}(\Omega_i)}(G_i)$. To show that $(\kappa^{-1})|\Omega_i$ normalises $H_2|\Omega_i$, let $j$ be such that $\Omega_j = \Omega_i^\kappa$. Then $\kappa$ witnesses the permutation isomorphism from $D_i$ to $D_j$ and maps $\alpha_i$ to $\alpha_j$, so $\kappa$ conjugates $H_2|\Omega_i = (D_i)_{(\alpha_i)}$ to $H_2|\Omega_j$. Therefore $(H_2|\Omega_i)^{\kappa^{-1}} = (H_2|\Omega_j)^{\kappa^{-1}} = H_2|\Omega_i$. Hence $(\kappa^{-1})|\Omega_i \leq N_i$ and so $i \in T$.

Part 2: Observe that $N_{\text{Sym}(\Omega_i)}(G_i) = \langle g_i, c_i \rangle$, so $N_i \leq \langle g_i \rangle \rtimes \langle c_i \rangle$. Assume for a contradiction that there exists a non-trivial $g \in \langle g_i \rangle$ and a $c \in \langle c_i \rangle$ such that $gc \in N_i$. Letting $r_i$ be the generator for $H_2|\Omega_i \cong C_2$, note that $r_i^{gc} = r_i$. Since $g$ moves $\alpha_i$ and $c$ fixes only $\alpha_i$, the points $\alpha_i^{gc}$ and $\alpha_i$ are distinct. But $(\alpha_i^{gc})^{r_i} = (\alpha_i^{gc})^{r_i^{gc}} = \alpha_i^{r_i^{gc}} = \alpha_i^{gc}$, a contradiction since $r_i$ only fixes $\alpha_i$, so $N_i \leq \langle c_i \rangle$.

To see that $N_i \geq \langle c_i \rangle$, first notice that $c_i \in N_{\text{Sym}(\Omega_i)}(G_i)$. Since $r_i$ is an element of $N_{\text{Sym}(\Omega_i)}(G_i)$ which fixes $\alpha_i$, it is in $\langle c_i \rangle$, and so $c_i$ normalises $H_2|\Omega_i$.

Lastly the isomorphism follows from Lemma 2.8.2. □

Fix a non-trivial orbit $\Omega_{11}$ of $H_2|\Omega_1$. For $2 \leq i \leq k$, let $\Omega_{i1} := \overline{\Omega_{11}}$ be an $(H_2|\Omega_i)$-orbit, and let $\Gamma = \bigcup_{i=1}^k \Omega_{i1}$. Then $H_2|\Gamma$ is in class $\mathfrak{P}(C_2)$, so $N_{\text{Sym}(\Gamma)}(H_2|\Gamma)$ can be computed using Algorithm 2.

Next, we show how $N_T(H_2)$ can be constructed from $N_{\text{Sym}(\Gamma)}(H_2|\Gamma)$. Define the map $\theta : N_{\text{Sym}(\Gamma)}(H_2|\Gamma) \to K$ by $\theta^i = \Omega_{i1} \in \Omega_j$, if $\Omega_{i1}^\theta = \Omega_{j1}$, and let the group $R = \langle N_{\text{Sym}(\Omega_1)}(G_1), \ldots, N_{\text{Sym}(\Omega_k)}(G_k), \text{Im}(\theta) \rangle$.

Lemma 7.4. $N_T(H_2) = \langle N_1, N_2, \ldots, N_k, \text{Im}(\theta) \rangle \leq R$.

Proof. Observe that as $|\langle H_2|\Omega_i \rangle| = 2$, the normaliser and the centraliser of $H_2|\Omega_i$ in $\text{Sym}(\Omega_i)$ coincide, so $N_i \leq N_T(H_2)$. To show that $\text{Im}(\theta) \leq N_T(H_2)$, let $g$ be an element
of $N_{Sym}(H_2|\Gamma)$ and let $h \in H_2$. Then there exists $h' \in H$ such that $(h|\Gamma)^g = h'|\Gamma$. We first show that $h^{\theta(g)} = h'$.

Fix $i$ and let $\Omega_j = \Omega_i^{\theta(g)}$, so $\Omega_j = \Omega_i^{\theta(g)}$. Then $(H|\Omega_i)^{\theta(g)} = H|\Omega_j$ as $\theta(g) \in K$. Since $K$ acts on the fixed points of $H_2$,

$$(H_2|\Omega_i)^{\theta(g)} = ((H|\Omega_i)_{\alpha_i})^{\theta(g)} \leq (H|\Omega_j)_{\alpha_j} = H_2|\Omega_j,$$

and equality is achieved since both groups have order two. Now $h|\Omega_i \neq 1$ if and only if $h'|\Omega_j$ and $(h|\Omega_i)^{\theta(g)}$ are the unique non-trivial element of $H_2|\Omega_j$, and $h|\Omega_i = 1$ if and only if $h'|\Omega_j = (h|\Omega_i)^{\theta(g)} = 1$. Therefore $(h^{\theta(g)})|\Omega_j = (h|\Omega_i)^{\theta(g)} = h'|\Omega_j$, as required.

So $\theta(g)$ normalises $H_2$, and as $\theta(g) \in K \leq T$, it follows that $\theta(g) \in N_T(H_2)$. Therefore $N_T(H_2) \geq \langle N_1, N_2, \ldots, N_k, \Im(\theta) \rangle$.

To show that these two groups are equal, let $\nu \in N_T(H_2)$ and let $\kappa := \theta(\nu|\Gamma) \in K \leq T$. Then as both $N_T(H_2)$ and $\Im(\theta)$ act on $O$, it follows that $\Omega_i^{\nu} = \Omega_j$ if and only if $\Omega_i^{\nu|\Gamma} = \Omega_j$ if and only if $\Omega_i^{\nu} = \Omega_j$. Therefore $\nu \kappa^{-1}$ is an element of $T$ which fixes each $\Omega_i$ setwise. By Lemma 7.3.2, $T \cong N_1 \wr S_k$, so $\nu \kappa^{-1} \in N_1 \times N_2 \times \cdots \times N_k$. $\square$

As in Theorem 4.10, we may compute $N_R(H_p)$ by considering all $\kappa \in \Im(\theta) \leq K$. Lastly, we show that $I$ can be computed from $N_R(H_p)$.

**Lemma 7.5.** Let $W \leq GL_k(p)$ be as in Section 3.1, let $\Xi : L \to W$ be as in Lemma 3.5 and let $\Xi|_T : T \to W$ be the restriction of $\Xi$ to $T$. Then $\Xi|_T$ is an isomorphism which maps $I$ to $\Xi(N_R(H_p))$.

**Proof.** It follows from Lemma 7.3.2 that $L = \langle T, G \rangle$. By Lemma 3.5.1, $\Xi$ is an epimorphism and $\Xi(G) = 1$, so $W = \Xi(L) = \Xi(T)$, and hence $\Xi|_T$ is an epimorphism. Note also that the groups $G \cong C^k_p$ and $T \cong C_{p-1} \wr S_k$ intersect trivially, so $L/G \cong T$. Hence $W \cong L/\ker(\Xi) \cong T$, therefore $\Xi|_T$ is an isomorphism.

By Lemma 7.3.1 and Lemma 7.4,

$$I = N_T(H_2) \cap N_T(H_p) \leq R \cap N_T(H_p) \leq N_R(H_p),$$

and so $\Xi(I) \subseteq \Xi(N_R(H_p))$. For the other containment, let $r \in N_R(H_p)$. Then the preimage $\tau := \Xi|_T^{-1}(\Xi(r))$ is an element of $T$ normalising $H_p$, so $\tau \in N_T(H_p)$.

Observe from Lemma 7.3.2 that $N_{Sym(\Omega_i)}(G_i) = \langle g_i, N_i \rangle$. So

$$R = \langle G, N_1, N_2, \ldots, N_k, \Im(\theta) \rangle,$$

which is $\langle G, N_T(H_2) \rangle$ by Lemma 7.4. Now as $\Xi(G) = 1$ by Lemma 3.5.1, $\Xi(R) = \Xi(N_T(H_2))$. This means that $\tau \in N_T(H_2)$ and so $\tau \in I$. Hence $\Xi(r) \in \Xi(I)$ and therefore $\Xi(N_R(H_p)) = \Xi(I)$. $\square$

Therefore, we compute $N_{S_n}(H)$ for $H \leq S_n$ in $\Im\mathfrak{P}(D_{2p})$ in the following way. First we construct the Sylow $p$-subgroup $H_p$ and a Sylow 2-subgroup $H_2$ of $H$. Then we find
Using Lemma 7.1, we generate $H$ as a small product of groups in $\mathfrak{I}_{n}\mathfrak{P}(C_2)$ and $\mathfrak{I}_{n}\mathfrak{P}(C_p)$. We generate these groups that are isomorphic to $C_{2}^{k/2}$ and $C_{p}^{k/2}$ respectively using the method described before. The rest of the experiment works the same as that for $\mathfrak{I}_{n}\mathfrak{P}(C_p)$. We report computation times in Fig. 2.

All algorithms are implemented in GAP, apart from `makeInvSetPartn` in Algorithm 3, which is implemented in C++.

---

Table 1

| $p$ | $s$ | FULLSEARCH | LIMITDEPTH |
|-----|-----|------------|------------|
| 5   | 4   | 0.11       | 0.125      |
| 5   | 6   | 0.4765     | 0.625      |
| 5   | 8   | 3.0625     | 2.953      |
| 5   | 10  | 8.2415     | 65.75      |

$N_{\text{Sym}}(\Gamma)(H_2|\Gamma)$ and $N_R(H_p)$ using Algorithm 2. Next we compute $\Xi^{-1}(\Xi(N_R(H_p)))$, which is equal to $N_L(H_p) \cap N_L(H_2)$ by Lemma 7.5. By Proposition 7.2, the normaliser $N_{S_n}(H)$ is equal to $\langle N_L(H_p) \cap N_L(H_2), H \rangle$.

8. Results

In our experiments we considered groups $H \leq S_{pk}$ in class $\mathfrak{I}_{n}\mathfrak{P}(C_p)$ that are isomorphic to $C_{p}^{k/2}$, for $p = 2, 3, 5, 11$. We generate these instances by populating the entries of a $k/2 \times k$ matrix with random elements of $F_p$. We rerun the generation if rank $M \neq k/2$.

For each value of $p$ and $k$, we create 10 instances of $H$ and compute $N_{S_n}(H)$ using both the GAP function `NORMALIZER` and our new algorithm, each run with a 10-minute time limit. We report the median, lower quartile and upper quartile time, in seconds, in Fig. 1. The lower and upper boundaries of the shaded area give the lower and upper quartiles respectively.

Next we compare the performance of computing $N_{S_n}(H)$ for $H \leq S_n$ in class $\mathfrak{I}_{n}\mathfrak{P}(C_p)$ using the methods described in Proposition 4.2 and Theorem 4.10 respectively. The results are shown in Table 1, where FULLSEARCH refers to the algorithm described in Section 6. To obtain complexity $2^{O(\frac{p}{2} \log \frac{p}{2} + \log n)}$, LIMITDEPTH is a combination of methods of Proposition 4.2 and Theorem 4.10. The algorithm is as follows. Let $H \in \mathfrak{I}_{n}\mathfrak{P}(C_p)$ have order $p^a$. As in Algorithm 2, we perform backtrack search in $K$. At a node at depth $s$, we iterate over all $(p-1)^s$ elements of $(\mathbb{F}_p^*)^s$, as in Proposition 4.2. If we succeed in finding a normalising element $g \in N_{S_n}(H)$, we update $N$ as $\langle N, g \rangle$, else we backtrack. Results (Table 1) show that even though FULLSEARCH has a higher worst case complexity, it performs much better than LIMITDEPTH in practice, especially where $p$ or $s$ are large.

We also consider $H \leq S_{pk}$ in class $\mathfrak{I}_{n}\mathfrak{P}(D_{2p})$ as in Section 7, for $p = 3, 11$. Using Lemma 7.1.1, we generate $H$ as product of groups in $\mathfrak{I}_{n}\mathfrak{P}(C_2)$ and $\mathfrak{I}_{n}\mathfrak{P}(C_p)$. We generate these groups that are isomorphic to $C_{2}^{k/2}$ and $C_{p}^{k/2}$ respectively using the method described before. The rest of the experiment works the same as that for $\mathfrak{I}_{n}\mathfrak{P}(C_p)$. We report computation times in Fig. 2.
Fig. 1. Median log time (secs) to compute $\mathcal{N}_{S_n}(H)$ for 10 instances of $H \leq S_n$ in $\text{InP}(C_p)$.

Fig. 2. Median log time (secs) to compute $\mathcal{N}_{S_n}(H)$ for 10 instances of $H \leq S_n$ in $\text{InP}(D_{2p})$. 
Acknowledgments

The first and third authors would like to thank the Isaac Newton Institute for Mathematical Sciences, Cambridge, for support and hospitality during the programme “Groups, Representations and Applications: New perspectives”, where work on this paper was undertaken. This work was supported by EPSRC grant no EP/R014604/1. This work was also partially supported by a grant from the Simons Foundation. The first and second authors are supported by the Royal Society (RGF\EA\181005 and URF\R\180015).

References

[1] L. Babai, Graph isomorphism in quasipolynomial time, in: Proc. 48th ACM SIGACT STOC, ACM, 2016, pp. 684–697.
[2] L. Babai, P. Codenotti, J.A. Grochow, Y. Qiao, Code equivalence and group isomorphism, in: Proc. 22nd ACM-SIAM SODA, SIAM, 2011, pp. 1395–1408.
[3] R.A. Beauregard, J.B. Fraleigh, A First Course in Linear Algebra, Houghton Mifflin Co., 1973.
[4] M.S. Chang, C. Jefferson, Disjoint direct product decompositions of permutation groups, J. Symb. Comput. 108 (2022) 1–16.
[5] M.S. Chang, C.M. Roney-Dougal, Primitive normalisers in quasipolynomial time, Arch. Math. 118 (2022) 19–25.
[6] T. Feulner, The automorphism groups of linear codes and canonical representatives of their semi-linear isometry classes, Adv. Math. Commun. 3 (4) (2009) 363–383.
[7] The GAP Group, GAP – Groups, Algorithms, and Programming, Version 4.11.0, 2020.
[8] D.F. Holt, The computation of normalizers in permutation groups, J. Symb. Comput. 12 (4–5) (1991) 499–516.
[9] D.F. Holt, B. Eick, E.A. O’Brien, Handbook of Computational Group Theory, Chapman & Hall/CRC, 2005.
[10] A. Hulpke, Constructing transitive permutation groups, J. Symb. Comput. 39 (1) (2005) 1–30.
[11] A. Hulpke, Normalizer Calculation Using Automorphisms, Contemp. Math., vol. 470, Amer. Math. Soc., 2008, pp. 105–114.
[12] E.M. Luks, Permutation groups and polynomial-time computation, in: DIMACS Ser. Discrete Math. Theoret. Comput. Sci., vol. 11, Amer. Math. Soc., 1993, pp. 139–175.
[13] E.M. Luks, T. Miyazaki, Polynomial-time normalizers for permutation groups with restricted composition factors, in: Proc. 2002 ISSAC, ACM, 2002, pp. 176–183.
[14] L. Miyamoto, An improvement of GAP normalizer function for permutation groups, in: Proc. 2006 ISSAC, ACM, 2006, pp. 234–238.
[15] E. Petrank, R.M. Roth, Is code equivalence easy to decide?, IEEE Trans. Inf. Theory 43 (5) (1997) 1602–1604.
[16] C.M. Roney-Dougal, S. Siccha, Normalisers of primitive permutation groups in quasipolynomial time, Bull. Lond. Math. Soc. 52 (2) (2020) 358–366.
[17] N. Sendrier, D.E. Simos, The hardness of code equivalence over $F_q$ and its application to code-based cryptography, in: Post-Quantum Cryptography, Springer, 2013, pp. 203–216.
[18] A. Seress, Permutation Group Algorithms, Cam. Tracts Math., vol. 152, CUP, 2003.
[19] J.H. van Lint, Introduction to Coding Theory, Grad. Texts Math., vol. 86, Springer-Verlag, 1999.
[20] D. Wiebking, Normalizers and permutational isomorphisms in simply-exponential time, in: Proc. 31st ACM-SIAM SODA, SIAM, 2020, pp. 230–238.