Computing with Group Homomorphisms

CHARLES R. LEEDHAM-GREEN†, CHERYL E. PRAEGER‡, AND LEONARD H. SOICHER†
† School of Mathematical Sciences, Queen Mary and Westfield College, Mile End Road, London E1 4NS, UK, and ‡ Department of Mathematics, University of Western Australia, Nedlands, Western Australia 6009
(Received 29 March 1989)

Let \( G = \langle X \rangle \) and \( H \) be finite groups and let \( \phi : X \to H \) be a map from the generating set \( X \) of \( G \) into \( H \). We describe a simple approach for deciding whether or not \( \phi \) determines a group homomorphism from \( G \) to \( H \), and if it does, for computing the kernel of \( \phi \). If \( G \) and \( H \) are permutation groups the algorithm is a simple application of standard algorithms for bases and strong generating sets. If \( G \) and \( H \) are soluble groups given in the usual way by PAG-systems and corresponding power conjugate presentations then the algorithm is a simple application of the non-commutative Gauss algorithm for constructing a subgroup of a soluble group. Further, a probabilistic algorithm is given for finding the kernel and image of \( \phi \) when each of \( G \) and \( H \) is given as a permutation group or a soluble group, \( |G| \) is known, and \( \phi \) is known to determine a homomorphism.

1. Introduction

We describe algorithms for computing with group homomorphisms. Let \( G = \langle X \rangle \) and \( H \) be finite groups and let \( \phi : X \to H \) be a map.

The first algorithm, given in section 2, is used to decide if \( \phi \) determines a homomorphism from \( G \) to \( H \), and if it does the kernel of \( \phi \) is computed. This algorithm, in the case where \( G \) and \( H \) are permutation groups, is a simple application of Sims' algorithms for bases and strong generating sets (see Leon, 1980 and Sims, 1971). If \( G \) and \( H \) are finite soluble groups given by PAG-systems and corresponding power conjugate (or power commutator) presentations (see Lane et al., 1984), then our algorithm is a simple application of the non-commutative Gauss algorithm (NCGA) of Newman (see Lane et al., 1984, pp. 105-110). In both these cases it is easy to compute images of elements of \( G \) and preimages of elements of \( \phi(G) \).

In section 3 we describe an efficient probabilistic algorithm for computing the kernel and image of \( \phi : G \to H \) given that \( \phi \) is a homomorphism and that \( |G| \) is known. This is closely related to the random Schreier algorithm (see Leon, 1980) in the case where \( G \) and \( H \) are permutation groups.

Our first algorithm has been implemented using the group theory language CAYLEY (Cannon, 1984) when \( G \) and \( H \) are permutation groups.

2. Homomorphisms

Let \( G = \langle X \rangle \) and \( H \) be finite groups. We note that any map \( \phi : G \to H \) can be identified with the subset

\[
s(\phi) = \{ (g, \phi(g)) \mid g \in G \}
\]
of $G \times H$, and that $\phi$ is a group homomorphism if and only if $s(\phi)$ is a subgroup of $G \times H$. We exploit this simple fact below. Suppose that we are given a map

$$\phi : X \to H$$

of the generating set $X$ of $G$ into $H$, and consider the subgroup

$$\Phi = \langle (x, \phi(x)) | x \in X \rangle$$

of $G \times H$. We wish to determine whether or not $\phi$ determines a homomorphism from $G$ to $H$, and if it does to compute its kernel. Theoretically the information we require is provided by the following proposition.

**Proposition 2.1.** (a) The mapping $\phi : X \to H$ determines a homomorphism from $G$ to $H$ if and only if the intersection $\Phi \cap (\{1_G\} \times H)$ is the identity subgroup of $G \times H$.

(b) If $\phi : G \to H$ is a homomorphism then its kernel $\ker \phi$ is the projection onto $G$ of the intersection $\Phi \cap (G \times \{1_H\})$.

**Proof.** (a) The map $\phi : X \to H$ determines a homomorphism from $G$ to $H$ if and only if each relation for $G$ is also a relation for $\phi(G)$, that is, if and only if for each word $w$ in $X \cup X^{-1}$ (where $X^{-1} = \{x^{-1} | x \in X\}$) which is equal to $1_G$ in $G$ we have $\phi(w) = 1_H$ in $H$. This in turn is true if and only if $\Phi \cap (\{1_G\} \times H)$ is the identity subgroup of $G \times H$.

(b) An element $g \in G$ lies in the kernel of $\phi$ if and only if $(g, 1, \phi) \in \Phi$.

If it is easier to calculate the orders of $G$ and $\Phi$ than to form the intersection for the homomorphism test, then the fact that $\phi$ determines a homomorphism if and only if $|G| = |\Phi|$ is used.

2.1. **The Permutation Group Case**

Suppose now that $G$ is a permutation group on the set $\{1, \ldots, n\}$ with a base $B$ (that is, $B$ is a sequence of distinct elements of $\{1, \ldots, n\}$ such that only the identity element of $G$ fixes $B$ pointwise), and that $H$ is a permutation group on the set $\{n+1, \ldots, n+m\}$ with a base $C$. Then $G \times H$ has a natural representation as a permutation group on the set $\{1, \ldots, n+m\}$ and has a base

$$D = C \text{ concatenate } B.$$

Now $\Phi \cap (\{1_G\} \times H)$ is the pointwise stabilizer of $B$ in $\Phi$, and $\Phi \cap (G \times \{1_H\})$ is the pointwise stabilizer of $C$ in $\Phi$. Thus we have:

**Corollary 2.2.** (a) The map $\phi$ determines a homomorphism of permutation groups if and only if $B$ is a base for $\Phi$, that is $\Phi \cap (\{1_G\} \times H) = \{1_G \times H\}$.

(b) If $\phi$ is a homomorphism then $\ker \phi$ is the pointwise stabilizer of $C$ in $\Phi$.

Our homomorphism algorithms for permutation groups can proceed as follows. First, calculate a strong generating set for $\phi$ with respect to its base $D$. If at any point in this computation we find that $|\Phi| > |G|$ then we can stop, with the knowledge that $\phi$ does not determine a homomorphism. If at the end of the strong generating set calculation we have that $|\Phi| = |G|$, then $\phi$ determines a homomorphism. Suppose now that this is the case. Then a strong generating set for the image $\Phi(G)$ with respect to its base $C$ can be
read off from the strong generating set $S$, say, for $\bar{\phi}$ by taking the set of permutations of 
$\{n+1, \ldots, n+m\}$ induced by $S$. Furthermore, a base for Ker $\phi$ is $B$, and a corresponding 
strong generating set for Ker $\phi$ consists of the permutations on $\{1, \ldots, n\}$ induced by the 
elements of $S$ which fix $C$. It is now also easy to find preimages of elements of $\phi(G)$. 
Given an element $h$ of $\phi(G)$, since $C$ is an initial segment of our base $D$ for $\bar{\phi}$, it is a 
standard calculation to determine an element $(g_0, h_0) \in \bar{\phi}$ such that $C^h = C^{(g_0, h_0)}$ (see 
Leon, 1980). Since $C$ is a base for $H$ we must have $h = h_0$, and so $\phi(g_0) = h$.

How do we compute images of elements of $G$? Given that $\phi$ is a homomorphism we 
know that the base $B$ for $G$ is also a base for $\phi$, and we can use the base change algorithm 
of Sims (1971) to calculate a strong generating set for $\phi$ with respect to $B$. Then given 
g \in G we can calculate the unique element $(g, h) \in \phi$ such that $B^h = B^{(g, h)}$, and we have 
that $\phi(g) = h$.

2.2. THE SOLUBLE GROUP CASE

Suppose that $G$ and $H$ are finite soluble groups given by PAG-systems $(g_1, \ldots, g_n)$ 
for $G$ and $(h_1, \ldots, h_m)$ for $H$, and corresponding power conjugate (or power commutator) 
presentations. (We prefer the use of power conjugate to power commutator presentations 
for collection multiplication of elements of non-nilpotent soluble groups (see Leedham-
Green & Soicher, 1990)). Then $G \times H$ has natural PAG-systems 

$$P_1 = (g_1, \ldots, g_n, h_1, \ldots, h_m) \quad \text{and} \quad P_2 = (h_1, \ldots, h_m, g_1, \ldots, g_n),$$

and corresponding power conjugate presentations instantly derivable from those for $G$ 
and $H$.

Our homomorphism algorithms for soluble groups proceed as follows. We take the 
PAG-system $P_2$ for $G \times H$, and use the NCGA to calculate the canonical generating 
system (CGS) (see Laue \textit{et al.}, 1984) for $\bar{\phi}$ with respect to $P_2$. If at any point in this 
computation we find that $|\bar{\phi}| > |G|$ then we can stop, with the knowledge that $\phi$ does not 
determine a homomorphism. If at the end of the NCGA calculation we have $|\bar{\phi}| = |G|$, 
then $\phi$ determines a homomorphism. Suppose now that this is the case. Then the CGS 
for $\bar{\phi}$ consists of elements of the form 

$$f_1^{\sigma_1(1)} \cdots g_n^{\sigma_1(n)} \quad (i = 1, \ldots, k), \quad \text{and} \quad g_1^{\sigma_k(1)} \cdots g_n^{\sigma_k(n)} \quad (i = k+1, \ldots, n), \quad (*)$$

where $(f_1, \ldots, f_k)$ is the CGS for $\phi(G)$ with respect to $(h_1, \ldots, h_m)$. We now note that 
the elements of $(*)$ with $i = k+1, \ldots, n$ form the CGS for Ker $\phi$ with respect to 
$(g_1, \ldots, g_n)$. It is also easy to find preimages of elements of $\phi(G)$. Since $(f_1, \ldots, f_k)$ 
is a CGS for $\phi(G)$, given $h \in \phi(G)$ it is straightforward (see Laue \textit{et al.}, 1984, p.109) to 
calculate an element of the form $(g, h) \in \phi$, and we have that $\phi(g) = h$.

How do we determine images of elements of $G$ given that $\phi$ is a homomorphism? We 
first calculate the CGS for $\bar{\phi}$ with respect to the PAG-system $P_1$ for $G \times H$. This 
is especially easy since we know that the elements of this CGS are of the form 

$$g_1^{\sigma(1)} \cdots h_n^{\sigma(n)} \quad (i = 1, \ldots, n).$$

Then given $g \in G$ it is a triviality to calculate the unique 
element of $\bar{\phi}$ of the form $(g, h)$, and we have that $\phi(g) = h$.

Our soluble group algorithms evolved from a recursive algorithm developed in Praeger 
(1987) for finding the kernel of a soluble group homomorphism. M. F. Newman suggested 
improvements in the original algorithm, and in particular noted that it could be modified 
to provide a PAG-system for the image with few assumptions about the codomain.
Newman also suggested that our present algorithms could be extended to the case of
polycyclic group homomorphisms. The kernel algorithm we describe here is similar to
the one given in the diploma thesis of Thiemann (1987).

3. The Random Algorithm for Kernels and Images

Suppose now that we are given a map \( \phi: X \rightarrow H \), where \( G = \langle X \rangle \) and \( H \) are finite
groups, and assume that \( \phi \) determines a homomorphism from \( G \) into \( H \) and that \( |G| \) is
known. Our random algorithm builds up the kernel \( \ker \phi \) and the image \( \text{Im} \phi \) of \( \phi \),
using elements of the kernel and image that we generate at random. In the case where
\( G \) and \( H \) are permutation groups, this algorithm is closely related to the random Schreier
algorithm of Leon (1980). We therefore expect that our random algorithm could be much
faster than the preceding deterministic algorithms for finding kernels and images.

Our algorithm relies on \( G \) and \( H \) being given in a form which makes the following
computations efficient. We must be able to build up an ascending chain of subsets of a
subgroup \( T \) of \( G \) or \( H \), starting with the set \( \{1_T\} \), given random elements \( t \) of \( T \). If \( S \) is
a set in this chain then we assume that \( |S| \) is known and divides \( |T| \), and furthermore
that we have a "generating" subset \( X(S) \subseteq S \), such that given \( t \in T \) there is an efficient
algorithm for determining if \( t \in S \), and if so, giving a word in \( X(S) \) representing \( t \). If \( t \not\in S \),
then we must be able to calculate a description of a superset \( \hat{S} \) of \( S \), such that \( t \in \hat{S} \subseteq T \),
\( |\hat{S}| \) is known and divides \( |T| \), and \( \hat{S} \) has a "generating" subset \( X(\hat{S}) \supseteq X(S) \) such that
every element in \( X(\hat{S}) \) is given by a word in \( X(S) \cup \{t\} \). Then \( \hat{S} \) is the next set in the
ascending chain.

For example, if \( G \) (or \( H \)) is a permutation group, then an appropriate description of
a subset \( S \) of a subgroup \( T \) is via a partial base and partial strong generating set for \( T \),
with the associated Schreier vectors (see Leon, 1980). The subset \( S \) of \( T \) being
described is a set product of the transversals defined by the Schreier vectors, and \( X(S) \)
is the partial strong generating set. The membership test and the algorithm to expand \( S \)
and \( X(S) \) given \( t \in S \) are as in the random Schreier algorithm (Leon, 1980, pp. 959-961).

If \( G \) (or \( H \)) is a soluble group given by a PAG-system and power conjugate presentation,
then an appropriate description of \( S \subseteq T \) is via a sequence \( (s_1, \ldots, s_n) \) of elements of \( T \)
in echelon form. The subset \( S \) of \( T \) being described is

\[
S = \{ s_1^{x(1)} s_2^{x(2)} \cdots s_n^{x(n)} \mid 0 \leq x(i) < p(i) \},
\]

where \( p(i) \) is the prime for which either \( s_i^{x(i)} = 1 \) or the weight (see Laue et al., 1984,
p. 106) of \( s_i^{x(i)} \) is greater than that of \( s_i \). The "generating" subset \( X(S) \) is simply \( \{s_1, \ldots, s_n\} \).
The membership test and the algorithm to expand \( S \) and \( X(S) \) given \( t \in S \) is given by
the algorithm (Laue et al., 1984, p. 109) to "insert an element into a subsequence of a CGS".

The other requirement of our algorithm is that we must be able to select an element
at random from the group \( G \), by which we mean that there should be a probability of
\( 1/|G| \) of a given element \( g \) of \( G \) being selected, and we must be able to calculate the
image \( \phi(g) \) of such a random element. In practice, however, "reasonably random"
elements are all that are needed. Leon (1980) suggests the following approach. Let \( r_0 = 1_G \),
and let the first "reasonably random" element \( r_1 \) be a random element of the generating
set \( X \) of \( G \). To form the next "reasonably random" element \( r_{n+1} \) from \( r_n \) we right multiply
\( r_n \) by a random element of \( X \) such that \( r_{n+1} \neq r_n \). This approach makes it easy to form
\( \phi(r_{n+1}) \) from \( \phi(r_n) \). A problem with Leon's method is that the generating set \( X \) is often
not a random collection of generators of \( G \). Often \( X \) consists of elements of low order,
and with pairs of elements of $X$ generating small subgroups of $G$. We have thus found it useful first to multiply pairs of elements $x, y \in X$ with $\text{order}(x) \leq \text{order}(y)$, and if the order of $xy$ is greater than that of $x$ then to replace $x$ in $X$ by $xy$ (and to keep a record of $\phi(xy)$). This process can be continued, usually producing a "more random" generating set $X$ containing higher order elements than before. Then we apply Leon's approach to generating "reasonably random" elements of $G$.

**DESCRIPTION OF THE RANDOM ALGORITHM**

**Input:**
(i) The groups $G = \langle X \rangle$ and $H$ have the computational properties above, and $|G|$ is known.
(ii) The map $\phi : X \rightarrow H$ determines a homomorphism.

**Output:**
The algorithm computes generating sets $X(K)$ and $X(I)$ for the kernel $K$ and image $I$ of $\phi$.

**The algorithm:**
(a) **Initial step.** We set $K = \{1_G\}$, $I = \{1_H\}$, and $X(K) = X(I) = \{\}$. (Throughout the algorithm $\tilde{X}(I)$ contains, for each $x \in X(I)$, exactly one $\tilde{x} \in G$ with $\phi(\tilde{x}) = x$.)
(b) **Test step.** If $|K| \cdot |I| = |G|$ then we stop, with $K$ and $I$ the kernel and image of $\phi$ respectively. Otherwise we go to the general step.
(c) **General step.** At this point we have descriptions for the subsets $K$ and $I$ of the kernel and image of $\phi$ respectively that we have found so far. Let $g$ be an element of $G$ chosen at random. If $\phi(g)$ is not in $I$ then we compute $\tilde{I}$ containing $I$ and $\phi(g)$ (as described above), and also a "generating" set $X(\tilde{I})$ containing $X(I)$. For each $x \in X(\tilde{I})$ we have an expression for $x$ as a word in $X(I) \cup \{\phi(g)\}$, and hence we can determine an $\tilde{x}$ such that $x = \phi(\tilde{x})$, and so form $\tilde{X}(\tilde{I}) = \tilde{X}(I)$. We set $I = \tilde{I}$, $X(I) = X(\tilde{I})$ and $\tilde{X}(I) = \tilde{X}(\tilde{I})$ and go to the test step.

If $\phi(g) \in I$ then we find an expression for $\phi(g)$ as a word $w$ in $X(I)$. Now $\tilde{X}(I)$ contains an element $\tilde{x} \in G$ with $x = \phi(\tilde{x})$, for each $x \in X(I)$, and we evaluate the word $w$ on the $\tilde{x}$ to obtain a $z \in G$ with $\phi(z) = \phi(g)$. Set $g^* = gz^{-1}$. If $g^* \in K$ then we replace $K$ by $\tilde{K}$ containing $K$ and $g^*$, and also replace $X(K)$ by $X(\tilde{K})$, and then go to the test step. If $g^* \in K$ then we have, unfortunately, achieved nothing in this step, and we do not change $K$ or $I$, but go to the general step again.

To see that the above algorithm really works, we show that the probability of leaving both $K$ and $I$ unchanged after a single run-through of the general step is at most $1/2$.

**Proposition 3.1.** The probability that both $K$ and $I$ are left unchanged after a single run-through of the general step is $|K| \cdot |I|/|G| \leq 1/2$.

**Proof.** We choose an element $g \in G$ at random. The probability $p$ that $K$ and $I$ are both left unchanged is the product $p_1 p_2$ where $p_1$ is the probability that $\phi(g)$ lies in $I$ and $p_2$ is the probability that $g^* \in K$ given that $\phi(g) \in I$. Now the preimage $\phi^{-1}(I) = \{x \in G | \phi(x) \in I\}$ is the union of $|I|$ cosets of the kernel $\text{Ker } \phi$ of $\phi$ and hence

$$p_1 = |\phi^{-1}(I)|/|G| = |I| \cdot |\text{Ker } \phi|/|G|.$$
If \( \phi(g) \in I \) then the algorithm determines an element \( z \) of \( G \), depending only on \( \phi(g) \), such that \( g^* = gz^{-1} \in \text{Ker} \phi \leq \phi^{-1}(I) \). Thus, for a given element \( \phi(g) \) of \( I \), each element of \( \text{Ker} \phi \) has equal probability of occurring as the element \( g^* \) (as \( g \) runs over \( \phi^{-1}(\phi(g)) = (\text{Ker} \phi)z \)). Thus, given that \( \phi(g) \in I \), the probability that \( g^* \in K \) is \( p_2 = |K|/|\text{Ker} \phi| \). Thus

\[
p = p_1p_2 = |I|/|K|/|G|.
\]

Finally we note that \( |K| \) divides \( |\text{Ker} \phi| \), \( |I| \) divides \( |\text{Im} \phi| \) and hence

\[
|K| \cdot |I| \text{ divides } |\text{Ker} \phi| \cdot |\text{Im} \phi| = |G|,
\]

so since \( |K| \cdot |I| < |G| \) we have \( p = 1/2 \).

We thank Mike Newman for his interest in this work and helpful discussions. The third author would like to thank the Universität Essen for its hospitality during which a version of this paper was prepared.

References


