AN ALGORITHM FOR DETERMINING DEFINING RELATIONS OF A SUBGROUP

by D. H. McLAIN

(Received 18 July, 1975)

1. Introduction. Suppose that G is a finitely presented group, and that we are given a set of generators for a subgroup H of finite index in G. In this paper we describe an algorithm by which a set of defining relations may be found for H in these generators.

The algorithm is suitable for programming on a digital computer. It appears to have significant computational advantages over the method of Mendelsohn [8] (which is based on the Schreier–Reidemeister results, see for example [4, pp. 91–95]) in those cases where the generators of H are given as other than the familiar Schreier–Reidemeister generators.

Both the algorithm and the proof of its sufficiency are based on the Todd–Coxeter coset enumeration process [9].

2. The algorithm. Suppose that G is a group generated by $g_1, \ldots, g_n$, with defining relations $R_1 = \ldots = R_r = 1$. Suppose that H is a subgroup of finite index in G, with generators $h_1, \ldots, h_m$. We may assume, without loss of generality, that the generators of H are among the generators of G, labelled so that

$$h_1 = g_1, \ldots, h_m = g_m, \quad \text{with} \quad n \geqq m. \tag{1}$$

For, suppose that one of the generators of H, say h_1, is given as a word $w(g)$ in the generators of G and their inverses, of length greater than 1. Then we may add h_1 to the list of generators of G and add the relation $w(g)h_1^{-1} = 1$ to the list of defining relations. This inclusion of the generators of H among those of G is an essential step of our algorithm; Mendelsohn [8] quotes an example to show that the relations $S_{j\lambda}(h) = 1$, defined below, are not sufficient to define H if this step is not taken.

Now suppose that the Todd–Coxeter–Benson–Mendelsohn procedure [1] is performed to determine the cosets of H in G. This procedure is an extension of the Todd–Coxeter coset enumeration process [9]. The Benson–Mendelsohn extension not only enumerates the cosets Hx_λ but also, at the same time, calculates the coset representatives x_λ as words in $g_1, \ldots, g_n$ and, for each coset representative x_λ and each generator g_i or its inverse g_i^{-1}, tabulates the product $x_\lambda g_i^\varepsilon$ in the form

$$x_\lambda g_i^\varepsilon = w(h)x_\mu, \tag{2}$$

where ε denotes ± 1, $w(h)$ represents a word in the generators $h_1, \ldots, h_m$ of H and x_μ is the representative of another (or possibly the same) coset. We shall refer to the formulae (2) as the T–C–B–M table.

For completeness, the procedure is summarised in the next paragraph, in a somewhat simplified form compared with the description in [1].

Starting from the table entries

$$x_1 = 1, \quad x_1 h_i^\varepsilon = h_i^\varepsilon x_1, \quad i = 1, \ldots, m, \quad \varepsilon = \pm 1, \tag{3}$$

which determine x_1 as the identity, we create, for any incomplete table entry $x_\lambda g_i^\varepsilon = ?$, a new coset representative $x_\mu = x_\lambda g_i^\varepsilon$ and we insert in the T–C–B–M table the two entries $x_\lambda g_i^\varepsilon = 1x_\mu$ and $x_\mu g_i^{-\varepsilon} = 1x_\lambda$. When we have enough such equations (2), we may calculate, by repeated application (as many times as the length of the word R), the product $x_\lambda R_i = w(h)x_\mu$; this differs from the original Todd–Coxeter process, which merely determines the coset μ, in that $w(h)$ is simultaneously formed by concatenating the appropriate H-words in the T–C–B–M table. If $\lambda \neq \mu$ then, as in the standard process, we must proceed to identify the two cosets, i.e. to force $\lambda = \mu$. The Benson–Mendelsohn extension requires us also to adjust the corresponding H-words in the table in the obvious way. Thus, for example, we must replace each occurrence of x_μ on the right-hand side of the table by $w(h)^{-1}x_\lambda$, by postmultiplying all the appropriate H-words by $w(h)^{-1}$, or alternatively replace each occurrence of x_λ by $w(h)x_\mu$. If, during this process of identifying the cosets μ and λ, the products $x_\lambda g_i^\varepsilon$ and $x_\mu g_i^\varepsilon$ have both been defined in the table, then we may have further identifications to make, again with the appropriate adjustments in the H-words in the table. The procedure is finished when every relation R_j has been applied to every coset λ to ensure that

$$\text{if} \quad x_\lambda R_j = w(h)x_\mu \quad \text{then} \quad \lambda = \mu. \tag{4}$$

Mendelsohn [6], [7] shows that the procedure does finish if $|G : H|$ is finite.

There are many possible modifications to the above, as to the basic Todd–Coxeter procedure, e.g. Leech [2], [3]. Although these are very useful in practice, to reduce the proliferation of cosets, they are not necessary for an understanding of the algorithm.

The method for finding the defining relations of H is as follows. The complete T–C–B–M table for the cosets of H in G allows us to express the product of a coset representative x_λ and any word $w(g) \in G$ in the form

$$x_\lambda w(g) = W(h)x_\mu, \tag{5}$$

where $W(h)$ is a word in $h_1, \ldots, h_m$. This is done by repeated applications of (2), one for each element g_i^ε in the word $w(g)$. In particular, we carry out this multiplication for every coset x_λ and every relation R_j, obtaining, in view of (4),

$$x_\lambda R_j = S_{j\lambda}(h)x_\lambda \tag{6}$$

for some word $S_{j\lambda}(h)$ in $h_1, \ldots, h_m$.

It is clear that in the group H the relations $S_{j\lambda}(h) = 1$ must hold. Conversely we shall show that these relations in the m generators $h_1, \ldots, h_m$ are sufficient to define H independently.

3. Example. Let G be the symmetric group on 3 objects generated by h, g subject to relations $R_1 = g^3$, $R_2 = hghg$, $R_3 = hg^2hg^2$, $R_1 = R_2 = R_3 = 1$ and let H be the group $\mathbf{Gp}\{h\}$. Then the T–C–B–M table is as follows.

https://doi.org/10.1017/S0017089500003025 Published online by Cambridge University Press

Coset representative	Generator or Inverse of Generator			
	h	h^{-1}	g	g^{-1}
$x_1 = 1$	hx_1	$h^{-1}x_1$	$1x_2$	$1x_3$
$x_2 = g$	$h^{-1}x_3$	hx_3	$1x_3$	$1x_1$
$x_3 = g^2$	$h^{-1}x_2$	hx_2	$1x_1$	$1x_2$

From this table we see that $x_2R_3 = h^{-2}x_2$ and $x_3R_2 = h^{-2}x_3$ and that the remaining seven combinations of λ, j all give $x_\lambda R_j = x_\lambda$. Thus the algorithm returns the one defining relation $h^{-2} = 1$ for the group H, which shows that H has order 2.

4. Proof of the algorithm.

THEOREM. *Let G be a group defined by the generators $h_1 = g_1, \ldots, h_m = g_m, g_{m+1}, \ldots, g_n$ and the relations $R_1 = \ldots = R_r = 1$. Let H be a subgroup of finite index, generated by $h_1, \ldots, h_m$, and let the Todd–Coxeter–Benson–Mendelsohn algorithm be performed on the cosets x_λ of H in G. If repeated use of the entries* (2) *in the T–C–B–M table enables each product $x_\lambda R_j$ to be expressed in the form*

$$x_\lambda R_j = S_{j\lambda}(h)x_\lambda, \tag{6}$$

for some word $S_{j\lambda}(h)$ in $h_1, \ldots,$ then H may be independently defined by the generators $h_1, \ldots, h_m$ and the relations $S_{j\lambda}(h) = 1$, for all relations R_j and cosets x_λ.

Proof. The proof consists, essentially, of a demonstration that no information is lost by adapting the Todd–Coxeter enumeration of the cosets of 1 in G (i.e. the elements of G) to make use only of the m generators of H, the relations $S_{j\lambda}(h) = 1$ and the information in the T–C–B–M table for $G : H$.

To give a rigorous proof, we distinguish between H, the subgroup of G, and the group defined by m generators and the relations $S_{j\lambda} = 1$ by calling the latter K. The generators of K will be $k_1, \ldots, k_m$ and if $w(h)$ is a word in $h_1, \ldots, h_m$ then $w(k)$ will denote the corresponding word in $k_1, \ldots, k_m$. We shall show that the mapping $w(h) \to w(k)$ is an isomorphism of H onto K.

Suppose first that G is finite, and consider the Todd–Coxeter process to enumerate the cosets of 1 in G. At each stage we have a set Γ of " cosets " $\alpha, \beta, \ldots$, which will end up as elements of G. We create a map f of Γ into the set of all pairs

$$(k, x_\mu),\ k \in K,\ 1 \leqq \mu \leqq |G : H|,$$

as follows. Start with $f(1) = (1, x_i)$. Whenever a new coset β is created from an existing one α by multiplying by g_i^ε, then $f(\beta)$ is defined to satisfy the condition:

$$\text{if } f(\alpha) = (w_1(k), x_\mu),\ \alpha g_i^\varepsilon = \beta \text{ and if the T–C–B–M table for}$$
$$G : H \text{ gives } x_\mu g_i^\varepsilon = w_2(h)x_\lambda, \text{ then } f(\beta) = (w_1(k)w_2(k), x_\lambda). \tag{7}$$

https://doi.org/10.1017/S0017089500003025 Published online by Cambridge University Press

We now prove that throughout the Todd–Coxeter process on $G:1$ the mapping remains well-defined and that (7) holds for every table entry $\alpha g_i^\varepsilon = \beta$. The next Todd–Coxeter step may be either (a) the creation of a new coset, (b) the identification of two cosets α, β when we find $\alpha R_j = \beta$, or (c) the identification of two cosets α, β from a previously identified pair $\gamma = \delta$ with $\alpha = \gamma g_i^\varepsilon$, $\beta = \delta g_i^\varepsilon$. We consider these separately.

(a) When a new coset β is created from α then the definition of $f(\beta)$ implies (7) for the entry $\alpha g_i^\varepsilon = \beta$. It also holds for the only other entry made at that time $\beta g_i^\varepsilon = \alpha$ because of the duality of the T–C–B–M table: $x_\mu g^\varepsilon = w(h)x_\lambda$ if $x_\lambda g^{-\varepsilon} = w(h)^{-1}x_\mu$.

(b) Suppose the Todd–Coxeter process finds $\alpha R_j = \beta$. Let $f(\alpha) = (w(k), x_\lambda)$ and $R_j = g_{i_1}^{\varepsilon_1} \dots g_{i_p}^{\varepsilon_p}$. Because the p table entries $\alpha g_{i_1}^{\varepsilon_1}$, $(\alpha g_{i_1}^{\varepsilon_1}) g_{i_2}^{\varepsilon_2}, \dots$ must be complete at this stage, by p-fold application of (7) and the definition (6) of $S_{j\lambda}(h)$, we conclude that

$$f(\beta) = (w(k)S_{j\lambda}(k), x_\lambda).$$

Because in the group K, $S_{j\lambda}(k) = 1$, we see that $f(\alpha) = f(\beta)$; so the mapping f remains well-defined after the identification $\alpha = \beta$.

(c) Suppose that in Γ, $\alpha = \gamma g_i^\varepsilon$, $\beta = \delta g_i^\varepsilon$, and that γ and δ have been successfully identified, so that $f(\gamma) = f(\delta) = (w_1(k), x_\lambda)$. Then (7) applies to both table entries γg_i^ε, δg_i^ε, so that

$$f(\alpha) = f(\beta) = (w_1(k)w_2(k), x_\mu),$$

where $w_2(h)$ and x_μ are given in the T–C–B–M table for $x_\lambda g_i^\varepsilon$. The identification $\alpha = \beta$ is thus consistent with f.

A simple induction argument on the number of steps in the Todd–Coxeter process shows that the relation (7) is true when, at the end of the process, the cosets α, β, ... have become the elements of the group G.

Now, repeated application of (7) shows that the set of elements of G whose f-images have the second component x_μ is just the coset Hx_μ. In particular, the set with second component x_1 forms H. If we identify the pair (k, x_1) with the element k of K, f gives a mapping of H into K. Because the generators h_i of H are among the elements g_i for which (7) holds, we have, for each generator h_i

$$f(h_i^\varepsilon) = f(1h_i^\varepsilon) = f(1)k_i^\varepsilon = k_i^\varepsilon,$$

and for each element h of H

$$f(hh_i^\varepsilon) = f(h)k_i^\varepsilon = f(h)f(h_i^\varepsilon).$$

It follows, by induction on the word length of elements of H, that f is both a homomorphism of H into K and the natural mapping $f(w(h)) = w(k)$. Since the generators of K are in $f(H)$, the homomorphism is onto K.

Further, as we have already remarked, the relations $S_{j\lambda}(h) = 1$ hold in H. Because these are the defining relations of K, the homomorphism f must be an isomorphism. This completes the proof of the theorem for finite G.

If G is infinite then the Todd–Coxeter process to enumerate the cosets α, β, ... of 1 in G will not, of course, terminate. However it can still be used as a tool in proving the theorem. We first show that the infinite Todd–Coxeter process can be a meaningful process. We must

ensure that for each coset α, and each generator and generator-inverse g_i^{ε} there will be a finite stage at which the coset αg_i^{ε} will be created, and also that, for each relation R_j, there will be a finite stage at which the condition $\alpha R_j = \alpha$ will be imposed. We must also ensure that when two cosets are identified $\alpha = \beta$, then it is the later one, β if $\beta > \alpha$, which will be deleted from the table. If the steps of the process are ordered to satisfy the above, then for any number N there will be a stage after which the entries for the first N cosets will never change. For suppose this holds for the first $N-1$ cosets. If we repeatedly identify the N-th coset with an earlier one, this would imply that all cosets ever created are among the first $N-1$ and hence that $|G:1| \leqq N-1$. Thus at some stage the N-th coset α is never identified with earlier ones (although, of course, we may never know when we have reached such a stage!); from then on the entries αg_i^{ε} can only decrease, and must eventually remain unchanged. The assertion follows by induction on N. By letting N tend to infinity we see that a limit exists, which may be taken as the result of the infinite Todd–Coxeter process.

The proof that this limit represents the multiplication table for the cosets of 1 in G is the same as for the finite case. The details are omitted, except for the remark that two words $w(g)$, $w^*(g)$ belong to the same coset of 1 in G if and only if there is a finite identity

$$w(g) = w^*(g)w_1(g)R_{i_1}^{\varepsilon_1}w_1(g)^{-1}\ldots w_s(g)R_{i_s}^{\varepsilon_s}w_s(g)^{-1}.$$

With this interpretation of the infinite Todd–Coxeter procedure, the proof of the theorem for finite G extends immediately to the case where G is infinite.

5. Computer implementation. The main difficulty in computer implementation of the Todd–Coxeter–Benson–Mendelsohn process and the present algorithm to determine defining relations of a subgroup is that some of the H-words in the T–C–B–M table can become rather long and rapidly fill the computer's store. Because of the large variance in the lengths of these words, the use of a language, such as Algol 68, [**10**], allowing variable array bounds may seem attractive. However the author has found it advantageous to handle a longer word as a list of other words, or their inverses. Thus the multiplication of two words (or in computing terms the concatenation of two strings) can be achieved by a subroutine which merely sets pointers to the two components, or, if one of them is the identity element, to the other component. Similarly the inverse of a word can be formed by setting a different pointer (e.g. negative) to the original word. If this approach is adopted, a "garbage collection" mechanism as implemented in most LISP compilers, e.g. [**5**], is desirable to determine which of the H-words are not components of others, and need not be retained. The whole process can readily be programmed in the common computer languages like Fortran which do not include pointers among their built-in facilities, since the subroutines which handle the pointers represent only a small proportion of the total.

In practical applications, where we may wish to repeat the algorithm on subgroups of subgroups, etc., it is necessary to follow the algorithm by another to prune at least most of the redundant relations. For example one might wish to use it, either in a single computer program or by calling a sequence of routines interactively from a terminal, to determine whether a finite group G is soluble; in theory the computer needs only to run the present method to determine a set of generators and relations for the derived group G', repeating

until the index of G' is equal to 1, and then to run the standard Todd–Coxeter process to find whether $|G:1| = 1$. Unfortunately there appears to be no practicable algorithm (i.e. a method guaranteed to find an optimal answer) to reduce a set of relations to a minimum set, so that heuristic techniques (i.e. ad hoc methods which perform reasonably well in most situations) are indicated. This is a further reason why the list method for word manipulation is advantageous, since an important constituent of such heuristic methods must be the search for frequently used common subexpressions.

REFERENCES

1. C. T. Benson and N. S. Mendelsohn, A calculus for a certain class of word problems, *J. Combinatorial Theory* **1** (1966), 202–208.

2. J. Leech, Coset enumeration on digital computers, *Proc. Cambridge Philos. Soc.* **59** (1963), 257–267.

3. J. Leech (ed.), *Computational problems in abstract algebra* (Pergamon, 1970).

4. W. Magnus, A. Karrass and D. Solitar, *Combinational group theory* (Interscience, 1966).

5. J. McCarthy, *Lisp 1.5 programmer's manual* (MIT Press, 1968).

6. N. S. Mendelsohn, An algorithmic solution for a word problem in group theory, *Canad. J. Math.* **16** (1964), 509–516.

7. N. S. Mendelsohn, Correction to: an algorithmic solution for a word problem in group theory, *Canad. J. Math.* **17** (1965), 505.

8. N. S. Mendelsohn, Defining relations for subgroups of finite index of groups with a finite presentation, in reference [3] above, 43–44.

9. J. A. Todd and H. S. M. Coxeter, A practical method for enumerating cosets of a finite abstract group, *Proc. Edinburgh Math. Soc.* (2) **5** (1936), 26–34.

10. A. Van Wijngaarden, B. J. Mailloux, J. E. L. Peck and C. H. A. Koster, Report on the algorithmic language Algol 68, *Numer. Math.* **14** (1969), 79–218.

COMPUTING CENTRE
UNIVERSITY OF SHEFFIELD
SHEFFIELD

PRINTED IN GREAT BRITAIN BY ROBERT MACLEHOSE AND CO. LTD
PRINTERS TO THE UNIVERSITY OF GLASGOW