1991

# Coset enumeration

Matthew T. Lazar
*San Jose State University*

Order Number 1344285

# Coset enumeration

Lazar, Matthew Thomas, M.S.

San Jose State University, 1991

U·M·I
300 N. Zeeb Rd.
Ann Arbor, MI 48106

# COSET ENUMERATION

A Thesis

Presented to

The Faculty of the Department of Mathematics

San Jose State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

By

Matthew T. Lazar

May, 1991

APPROVED FOR THE DEPARTMENT OF MATHEMATICS

Dr. Roger Alperin

Dr. Richard Kubelka

Dr. Brian Peterson

APPROVED FOR THE UNIVERSITY

ABSTRACT

COSET ENUMERATION

by Matthew T. Lazar

This thesis discusses the algorithm developed by Coxeter and Todd in 1936 called coset enumeration. We are interested in determining the index of a subgroup of a finitely presented group. Let F be a free group with generators $x_1, \ldots, x_n$. Let $r_1, \ldots, r_s$ be words in F, let R be the set of all conjugates in F of these words, and let [R] be the subgroup generated by R. Define K as F/[R] and $y_i$ as the image of $x_i$ under the canonical map. Let $g_1, \ldots, g_t$ be elements of F, $h_1, \ldots, h_t$ their images in K, and H the subgroup generated by the $h_i$. It is required to find the index of H in K.

## ACKNOWLEDGEMENTS

I would like to thank the members of my examining committee for their patience in reviewing this thesis. I would like to thank Dr. Roger Alperin for suggesting this thesis topic, and for his thoughtful comments concerning the presentation of it. Thanks also goes to my family and friends for their encouragement in writing this thesis.

TABLE OF CONTENTS

# INTRODUCTION

We are concerned with an algorithm for determining the index, when it is finite, of a subgroup H of a group K, when K is specified by a finite set of generators and relations and H is specified by a finite set of words in the generators of K. A systematic computational attack on the problem was discovered by Coxeter and Todd in 1936 and has proved to be a very useful tool in problems involving generators and relations in groups. Although the method was not completely formalized, it was evidently possible to convert it into a computer program, and this has been done by a number of people.

In Chapter I we introduce the basic concepts of free groups, presentations of groups, and Tietze transformations, which are needed as background to the understanding of the problem. Chapter I also gives a proof of the Nielsen-Schreier theorem, which states that

a subgroup of a free group of finite rank is again a free group.

In Chapter II, we give a formal statement of the coset enumeration problem, a formal description of the algorithm to solve the problem, and a proof that the algorithm works. We interpret each step of the algorithm in group-theoretic terms, and we are thus able to describe in such terms the point at which the algorithm terminates.

Chapter II continues with a modification by Leech of the coset enumeration algorithm. Enhanced coset enumeration allows us to express elements in the subgroup H of the group K in terms of the generators of H. Chapter II concludes with various implementations of the coset enumeration algorithm along with a discussion of some inherent limitations of the algorithm.

Finally, Chapter III discusses a second problem related to coset enumeration. Namely, we are given a finite presentation for a group K and a finitely generated subgroup of finite index whose generators are expressible in terms of the generators of K. Our problem is to find a presentation for H in terms of the generators of H. The solution to

this problem brings together the topics that were discussed in Chapters I and II, namely, the Nielsen-Schreier theorem, Tietze transformations, and the enhanced coset enumeration method of Leech.

It should be noted that the results presented in this thesis are by no means original. Mainly, this thesis is a gathering of various sources compiled into one paper which discusses the solutions to the two problems mentioned above.

In Chapter I, Preliminaries, Section 1.1, Free Groups, are discussed in Macdonald [7]. Sections 1.2, The Nielsen-Schreier Theorem, 1.3, Free Presentations of Groups, and 1.4, Tietze Transformations, are discussed in Johnson [4].

In Chapter II, Coset Enumeration, Sections 2.1, Statement of the Problem, and 2.2, The Algorithm, are covered in Trotter [11]. Section 2.4, Enhanced Coset Enumeration (Leech), is covered in Leech [5]. Sections 2.5, Different Implementations of Coset Enumeration, and 2.6, Limitations of Coset Enumeration, are covered in Cannon, Dimino, Havas, and Watson [1]. Section 2.7, Original Todd-Coxeter Coset

Enumeration, is covered in Johnson [4].

Finally, in Chapter III, Presentations of Subgroups, Sections 3.1, Statement of the Problem, and 3.2, The Method, are presented in Johnson [4]. The examples given in Sections 2.3 and 3.3 are original, in addition to the computer implementation as described in Section 2.8.

# CHAPTER I

# PRELIMINARIES

This chapter contains preliminary theorems needed for the discussion of coset enumeration in chapters two and three. First we define the concept of a free group with free basis. In the next section, we prove the Nielsen-Schreier theorem which states that a subgroup of a free group is again free. In section 1.3 we discuss free presentations of groups. Finally in section 1.4 we discuss Tietze transformations.

## 1.1 FREE GROUPS

We take any non-empty set X and proceed to define the free group on X. Let $X = \{x_\pi : \pi \in \Pi\}$ where $\Pi$ is a suitable index set. We take another set in one-one correspondence with X; call it $X^{-1}$ and

write its elements as $x_\pi^{-1}$, for $\pi \in \Pi$.

1.1.1 Definition - A word in the elements of $X \cup X^{-1}$ is an ordered set of n elements from $X \cup X^{-1}$, with repetitions allowed, for some $n \geq 0$. The length of the word is the integer n.

A typical word of length n>0 will be written $x_{\pi_1}^{e_1} \ldots x_{\pi_n}^{e_n}$ where $e_i = 1$ or -1. The unique word of length 0 will be 1; this word is called the "empty word." ($x^1$ means x.) As a word is an ordered set, it could be regarded as a vector.

As a consequence of the definition, two words are equal if and only if they have the same length and their corresponding terms are equal.

Next we define the product of two words: Let w be an arbitrary word. The product of 1 with w and w with 1 is w.

Let $u = x_{\pi_1}^{e_1} \ldots x_{\pi_n}^{e_n}$, $v = x_{\mu_1}^{d_1} \ldots x_{\mu_m}^{d_m}$. The product uv is defined to be the word $x_{\pi_1}^{e_1} \ldots x_{\pi_n}^{e_n} x_{\mu_1}^{d_1} \ldots x_{\mu_m}^{d_m}$. The length of the product is n+m and $uv \neq vu$ in general. The elements of $X \cup X^{-1}$ are words of length 1, and every word, except the empty word, is the product of certain

types of these particular words. The associative law holds and the set of words forms a monoid.

Now we define a group whose elements are certain equivalence classes of these words.

Let $W(X) = \{$ words in the elements of $X \cup X^{-1}\}$.

1.1.2. Definition - Two words $u,v \in W(X)$ are adjacent if there are words $z_1$, $z_2$ and an element $a \in X \cup X^{-1}$ for which $u = z_1 z_2$, $v = z_1 a a^{-1} z_2$; or $u = z_1 a a^{-1} z_2$, $v = z_1 z_2$. (Interpret $(x^{-1})^{-1}$ as x.)

1.1.3. Definition - Two words $u,v \in W(X)$ are equivalent if either $u = v$ or if there exist $z_1, z_2, \ldots, z_n$ with $n \geq 2$ such that $u = z_1$, $v = z_n$ and $z_i$ is adjacent to $z_{i+1}$ for $i = 1, \ldots, n-1$.

We can now verify that this provides an equivalence relation ~ on the set of words in W(X).

We have $u \sim u$ since $u = u$. If $u \sim v$ and $u \neq v$ then $u = z_1$, $v = z_n$ and $z_i$ is adjacent to $z_{i+1}$ for $i = 1, \ldots, n-1$. But then $v = z_n$, $u = z_1$ and $z_{i+1}$ is adjacent to $z_i$ for $i = n-1, \ldots, 1$, since adjacency is

symmetric. So v~u. If u~v and v~w then $u = z_1$ ,$v = z_n$ and $z_i$ is adjacent to $z_{i+1}$ for i = 1, . . ., n-1. Also $v = z_n = y_1$, $w = y_m$ and $y_i$ is adjacent to $y_{i+1}$ for i = 1, . . ., m-1. So $u = z_1$, $w = y_m$, and $z_i$ is adjacent of $z_{i+1}$ for i = 1, . . ., n-1; $z_n = y_1$, and $y_i$ is adjacent to $y_{i+1}$ for i = 1, . . ., m-1. So u~w.

We form the set of equivalence classes. The equivalence class containing w will be denoted by [w].

1.1.4 Definition - The *product* of equivalence classes [u], [v] of words in W(X) is defined to be [uv].

1.1.5 Theorem - The product of two equivalence classes of words in W(X) is well defined, and the set of all equivalence classes with this binary operation forms a group.

Proof:

The statement about being well defined is that [u'v'] = [uv] if [u'] = [u] and [v'] = [v]. The fact that u and u' are equivalent implies that [u'v] = [uv] because u'v and uv are adjacent if u' and u are adjacent. Similarly, [u'v'] = [u'v] follows from the equivalence of v and v'.

Therefore [u'v'] = [uv] as required.

Now we verify that we have a group. Closure is clear from the definition of product and subsequent justification.

Since [u]([v][w]) = [u][vw] = [u(vw)] = [(uv)w] = [uv][w] = ([u][v])[w], the associative law holds. The class of words containing 1 is an identity element as [w] [1] = [w] = [1][w]. If $w = a_1 a_2 \ldots a_n$ for $a_i \in X \cup X^{-1}$ then $[w][a_n^{-1} \ldots a_1^{-1}] = [1] = [a_n^{-1} \ldots a_1^{-1}][w]$. ///

1.1.6 Definition - The *free group on the* (nonempty) *set X* is the set of equivalence classes of words in W(X) with the binary operation described above.

1.1.7 Definition - A word in $X \cup X^{-1}$ is *reduced* if it has the form $x_{\pi_1}^{\varepsilon_1} \ldots x_{\pi_n}^{\varepsilon_n}$ with $x_{\pi_{i+1}}^{\varepsilon_{i+1}} \neq x_{\pi_i}^{-\varepsilon_i}$ for $i = 1, \ldots, n-1$.

($x^{-(-1)}$ is understood as x.)

1.1.8 Theorem - Each equivalence class of words in W(X) contains one and only one reduced word.

Proof:

Let $w \in W(X)$. There is no particular trouble in showing that if w is not reduced then it is adjacent to a word of smaller length; and

so an inductive argument will produce an equivalent reduced word.

To show that [w] contains only one reduced word requires more thought. We define a particular method of reduction, that is of obtaining a reduced word equivalent to a given word.

Let $w = a_1 \ldots a_n$ where $a_i \in X \cup X^{-1}$ for $1 \leq i \leq n$. Let $w_0 = 1$ and $w_1 = a_1$. Suppose $w_i$ is defined, where $1 \leq i < n$; we produce the definition for $w_{i+1}$ in two separate cases. Remember $w_i$ is a word, so that one can speak of its last term without ambiguity. If $w_i$ does not have a last term $a_{i+1}^{-1}$, then we put $w_{i+1} = w_i a_{i+1}$; if $w_i$ does end in $a_{i+1}^{-1}$, say $w_i = za_{i+1}^{-1}$, then z is uniquely determined ( because $z_1 a_{i+1}^{-1} = z_2 a_{i+1}^{-1}$ implies $z_1 = z_2$ by the definition of a word) and we define $w_{i+1}$ to be z.

This gives an inductive definition for $w_0, w_1, \ldots, w_n$. If n=0, $w_n$ is defined to be 1. A consequence of the definition is that $w_i$ is reduced for $0 \leq i \leq n$, and that in particular $w_n$ is reduced. Another fact

is that $w_i$ is equivalent to $a_1 \ldots a_i$ for each i and so $[w_n] = [w]$. Note that if w is already reduced, then $w_n = w$.

Next we show that two adjacent words u and v have reduced forms which are identical. Let $u = a_1 \ldots a_r\, a_{r+1} \ldots a_n$ and $v = a_1 \ldots a_r\, x\, x^{-1}\, a_{r+1} \ldots a_n$ where $x \in X \cup X^{-1}$. This choice of u and v is general enough. Suppose the procedure above gives the sequence $u_0=1, u_1, \ldots, u_n$ for u, $v_0=1, v_1, \ldots, v_{n+2}$ for v. Since $w_i$ was determined in the inductive definition by the first i factors in w, we have $u_0=v_0, u_1=v_1, \ldots, u_r=v_r$. We shall show that $u_r = v_{r+2}$ in two separate cases.

i) If $u_r$ does not end in $x^{-1}$ then $v_r = u_r$, $v_{r+1} = v_r x$, and $v_{r+2} = v_r$. (Here we use the definition of $w_i$.) Therefore $u_r = v_{r+2}$.

ii) If $u_r$ does end in $x^{-1}$, say $u_r = zx^{-1}$, then z cannot have a reduced form $z_0 x$ for any $z_0$; if it did then $u_r = z_0 x\, x^{-1}$ would not be in reduced form. Therefore $v_r = u_r$, $v_{r+1} = z$, and $v_{r+2} = zx^{-1} = u_r$ as required.

Therefore in either case we have $u_r = v_{r+2}$.

Since the final n-r factors in the expressions for u and v correspond when taken in the obvious order, we have $u_{r+i} = v_{r+2+i}$ for $i = 0, \ldots, n-r$. In particular, $u_n = v_{n+2}$, and it follows that the procedure for giving a reduced form yields the same result for u and v.

Finally, let u , v be any two reduced words in [w]. Because they are equivalent, they can be associated by a chain of adjacent words. We apply our procedure to all the words in the chain. The result is the same reduced word in each case. Since the procedure does not alter words which are already reduced, it will not alter u and v. Therefore u = v and u and v are identical. Therefore [w] contains precisely one reduced word. ///

We do not always write the elements of the free group on X as equivalence classes - we shall often merely use a representative from each class, usually the reduced word in the class, as no real confusion will result. We write [x] as x for $x \in X \cup X^{-1}$ and [1] as 1.

If the sets X and Y are in 1-1 correspondence, the free groups on

them are (clearly) isomorphic. It is also true that if two free groups are isomorphic then the corresponding sets X and Y have the same cardinality ( in other words, they are in 1-1 correspondence); we shall prove this for finite X and Y.

Thus the cardinality of X is an invariant of the free group F on X, and it is called the rank of F.

1.1.9 Definition - A *free basis* (or a set of free generators) for the free group F is a set of generators for F with the property that the only reduced word in them equal to 1 is the empty word. ( Of course, a free basis is said to generate F freely.)

A free group can have many bases other than the set X on which it was defined. It should be clear what the rigorous interpretation of the statement that a free group "has no relations" is. It is simply the fact that if $a_1 \ldots a_n$ is a reduced word equal to 1, with each $a_i \in X \cup X^{-1}$, then n = 0 and the word is empty.

For example, consider the free group F on the set X={x,y}. The elements x,y, which certainly generate F, have the property that any reduced word in them which equals 1 must be the empty word. But

other generators also have that property. Consider for instance $\{x,x^{-1}yx\}$. This set generates a subgroup of F which is F itself because it contains both x and y. A reduced word in x and $x^{-1}yx$ is the conjugate of a reduced word of the same length in x and y; for example $x^{-1}(x^{-1}yx)(x^{-1}yx)x\,x\,(x^{-1}yx)^{-1} = x^{-1}\,(\,x^{-1}y\,y\,x\,x\,y^{-1})\,x$. Therefore if w' = 1 where w' is a reduced word in $\{x, x^{-1}yx\}$, then $x^{-1}$ w x = 1 where w is a reduced word in {x,y} of the same length as w', and so w =1, whence w is the empty word, so w' is the empty word.

Using the same group F which is free on {x,y}, we have that {x,y,xy} is a set of generators which is not a basis. Also, the length of the word depends on the free basis in use; thus $x^{-1}yx$ has length 3 when referred to {x,y} in the above example, and length 1 when referred to $\{x,x^{-1}yx\}$.

1.1.10 Theorem - Let F be a free group with free basis $X = \{x_\pi: \pi \in \Pi\}$ and let G be an arbitrary group. If $\{g_\pi: \pi \in \Pi\}$ is a set of arbitrary elements of G, then there exists a unique homomorphism from F into G which maps $x_\pi$ to $g_\pi$ for all $\pi \in \Pi$.

Proof:

Let $\Phi$ be the mapping from X to G for which $\Phi(x_\pi) = g_\pi$ for all $\pi \in \Pi$. Let [w] be any element of F. We suppose $w = x_{\pi_1}^{e_1} \ldots x_{\pi_n}^{e_n}$, and define $\Phi([w])$ to be $g_{\pi_1}^{e_1} \ldots g_{\pi_n}^{e_n}$. In order to show $\Phi$ is a mapping we have to show that if [u] = [v] then $\Phi$ ([u]) = $\Phi$ ([v]). But this is clear if u and v are adjacent, and therefore it holds when they are equivalent.

Let [u], [v] $\in$ F so [u][v] = [uv]. Then $\Phi$( [uv]) = $\Phi$ ([u]) $\Phi$ ([v]) by the above definition and $\Phi$ ([u][v]) = $\Phi$ ([u]) $\Phi$ ([v]) as required. Since $\Phi$ maps $[x_\pi]$ to $g_\pi$, it is the required homomorphism. Its uniqueness follows from the fact that the images of the reduced words in W(X) are determined once the images of the $x_\pi$ are specified.

///

1.1.11 Theorem - Let $F_m$ and $F_n$ be free groups of finite ranks m and n respectively. Then $F_m$ is isomorphic to $F_n$ if and only if m = n.

Proof:

Suppose m = n. We mentioned before that it is clear that $F_m$ is isomorphic to $F_n$.

Conversely, suppose $F_m$ is isomorphic to $F_n$, and let G denote a group of order 2 with generator g. Suppose $F_m$ has free basis $\{x_1, \ldots, x_m\}$ and consider a homomorphism from $F_m$ to G. Such a homomorphism is determined uniquely by the images of each $x_i$, and each $x_i$ may map to g or 1; there are therefore $2^m$ -1 distinct homomorphisms from $F_m$ onto G. The kernel K of a homomorphism from $F_m$ onto G is a normal subgroup of $F_m$ and $F_m/K$ is isomorphic to G. Conversely, a normal subgroup of index 2 in $F_m$ is the kernel of a homomorphism of $F_m$ onto an isomorphic copy of G. It follows that $F_m$ has precisely $2^m$ -1 normal subgroups of index 2. Similarly $F_n$ has precisely $2^n$-1 normal subgroups of index 2. Because $F_m$ is isomorphic to $F_n$, we have $2^m$-1 = $2^n$-1, and so m = n as required. ///

1.1.12 Corollary - Suppose the free group F has a free basis of finite order n. Then every finite free basis for F has precisely n elements.

Proof:

Let $\{x_1, \ldots, x_n\}$ be a free basis for F. Then F is isomorphic to $F_n$. Suppose $\{y_1, \ldots, y_m\}$ is another free basis for F. Then F is isomorphic to $F_m$, whence $F_m$ is isomorphic to $F_n$, and so m = n. ///

Therefore if the number of elements in a free basis for a free group F is finite, then this number is well defined.

We now give a converse to the theorem stating the existence of homomorphisms from a free group into an arbitrary group. As a matter of convenience we define the free group on the empty set to be the group with one element.

1.1.13 Theorem - If G is a group with generating set $\{ g_\pi: \pi \in \Pi\}$ which has the property that for an arbitrary group H containing the subset $\{h_\pi: \pi \in \Pi\}$, the mapping $\Phi: g_\pi \rightarrow h_\pi$ can be extended to a homomorphism of G into H, then G is a free group and $\{g_\pi: \pi \in \Pi\}$ is a free basis for G.

Proof:

We make a particular choice for H. Take H to be the free group with $\{h_\pi: \pi \in \Pi\}$ as a free basis. There is by Theorem 1.1.10 a unique homomorphism $\Phi'$ from H to G with $\Phi'(h_\pi) = g_\pi$ for all $\pi \in \Pi$. We are given, in addition, a homomorphism $\Phi$ from G to H with $\Phi(g_\pi) = h_\pi$. But then $(\Phi \circ \Phi')(h_\pi) = h_\pi$ and $(\Phi' \circ \Phi)(g_\pi) = g_\pi$ and so $\Phi \circ \Phi'$ and $\Phi' \circ \Phi$ are the identity mappings in H, G respectively. It follows that $\Phi$ is an isomorphism. Since H is free with basis $\{h_\pi: \pi \in \Pi\}$, it follows that G is free with basis $\{g_\pi: \pi \in \Pi\}$. ///

Sometimes a free group is defined by the property mentioned in this theorem, but we have preferred a more constructive definition in the present account.

## 1.2 THE NIELSEN-SCHREIER THEOREM

We now embark on the proof of the fundamental theorem of combinatorial group theory -- the Nielsen-Schreier Theorem. This

theorem asserts that all the subgroups of a free group are again free. While we prove the result only for free groups of finite rank (this is all we need), the proof easily extends to the general case granted the postulate that any set can be well ordered, which is a version of the Axiom of Choice. The proof is divided into five steps. We fix the following notation:

Let $S = \{s_1, \ldots, s_r\}$ and let r = the rank of the free group F.

Let $T = S \cup \{ s_i^{-1} : 1 \leq i \leq r \}$.

Step 1. The ordering of F.

The elements of F are reduced words of the form $w = x_1 \ldots x_n$ where $x_i \in T$, $x_i x_{i+1} \neq 1$, n is the length of the word, $n = l(w)$, and take $l(1) = 0$. Order the elements of T as $s_1 < s_2 \ldots < s_r < s_1^{-1} < s_2^{-1} \ldots < s_r^{-1}$. If v, w are elements of F, define $v < w$ if $l(v) < l(w)$ and then order words of equal length lexicographically, that is, if $v = x_1 \ldots x_n \neq w = y_1 \ldots y_n$, $x_i, y_i \in T$, and m is the least subscript such that $x_m \neq y_m$, then $x_m < y_m$ implies

$v < w$ and $x_m > y_m$ implies $v > w$. This is a total ordering and even a well ordering.

1.2.1 Lemma - Let $w = x_1 \ldots x_n$, $x_i \in T$, $n \geq 1$, be a reduced word in F. Then for $v \in F$ we have $v < x_1 \ldots x_{n-1}$ implies $vx_n < w$.

Proof:

If $l(v) < n-1$, then $l(vx_n) \leq l(v) + 1 < n = l(w)$. Otherwise, $v = y_1 \ldots y_{n-1}$, $y_i \in T$, there is a least m such that $y_m \neq x_m$, and $y_m < x_m$. If $y_{n-1} x_n = 1$, then $l(v) = n-2 < n = l(w)$ and we are done. If $vx_n = y_1 \ldots y_{n-1} x_n$ is reduced, since $l(vx_n) = l(w)$, $y_1 = x_1, \ldots, y_{m-1} = x_{m-1}$, and $y_m < x_m$, we have $vx_n < w$ as required. ///

Step 2. The Schreier transversal.

We fix a subgroup H of F.

The right cosets of H in F yield a partition of F, and (by the axiom of choice) we can find a subset U of F such that for any $x \in F$, there is exactly one element $u \in U$ such that $x \in Hu$.

Such a subset U is called a (right) *transversal of H in F*. The

*Schreier transversal* with respect to the previous ordering is obtained if the representative in U of each coset is taken to be the least element of that coset.

We list the cosets Hx as x runs over F in ascending order; thus $H \cdot 1, Hs_1, Hs_2, \ldots$, and for each $x \in F$, delete Hx from the list if $Hx = Hy$ for some $y < x$. Put U = the set of x such that x remains on the list. By construction, the transversal U has the property that $x<y$, and $Hx = Hy$ together imply that $y \notin U$. Note that since F is a free group of finite rank, it contains only countably many reduced words. There are only a finite number of reduced words of a given length. Therefore, the above construction is possible.

The following is called the Schreier property:

1.2.2 - Lemma - Let $x_1 \ldots x_n$ be a reduced word in F ($n \geq 1$); then $x_1 \ldots x_n \in U$ implies $x_1 \ldots x_{n-1} \in U$.

Proof:

Suppose $x_1 \ldots x_{n-1} \notin U$. Then there exists a $u \in U$ such that $Hu = Hx_1 \ldots x_{n-1}$. Therefore $u < x_1 \ldots x_{n-1}$. By Lemma 1.2.1

$ux_n < x_1 \ldots x_n$. Let $v \in U$ such that $Hv = Hux_n$. Therefore $v \leq ux_n$.

Then $v < x_1 \ldots x_n$ and $Hv = Hx_1 \ldots x_n$ so $x_1 \ldots x_n \notin U$ as required. ///

Step 3 - The subset A of H.

Let $u \in U$ , $x \in T$. As $ux \in F$ and U is a transversal for H in F, there is one $v \in U$ such that $ux \in Hv$. Since v depends on u and x, we denote it by $\tau(ux)$. Since $ux = h\,\tau(ux)$ for some $h \in H$, we have $ux\,(\tau(ux))^{-1} = h \in H$ for all $u \in U$ and $x \in T$.

Put $A = \{ ux(\tau(ux))^{-1} : u \in U , x \in T\}$, which is a subset of H.

1.2.3 Lemma - The set A just defined generates H.

Proof:

Let $x \in H$. We can write $x = x_1 \ldots x_n$, a reduced word with $x_i \in T$. Let $u_1 = 1$, $u_{i+1} = \tau(u_i x_i)$, $i = 1, \ldots, n$. Consider the sequence $a_i = u_i x_i u_{i+1}^{-1}$ for $1 \leq i \leq n$. By definition, each $a_i \in A$ and so H contains $a_1 \ldots a_n = u_1 x_1 \ldots x_n u_{n+1}^{-1} = x\, u_{n+1}^{-1}$. Since $x\, u_{n+1}^{-1} = h \in H$, $u_{n+1} = h^{-1}x \in H$, since $x \in H$, and also

$u_{n+1} \in U$; therefore $u_{n+1} = 1$, since 1 is the unique element of U representing the trivial coset H. Therefore $x = a_1 \ldots a_n$, proving that A generates H, as claimed. ///

Step 4. Further properties of A.

1.2.4 Lemma

i) $ux\,(\tau\,(ux))^{-1} = 1$ if and only if $ux \in U$ for $u \in U$ and $x \in T$.

ii) $u = \tau\,(\tau\,(ux)x^{-1})$ for all $u \in U$ and $x \in T$.

iii) Let $ux, vy \in UT\backslash U$; then either

a) $x(\tau\,(ux)^{-1})vy = 1$ in which case $v = \tau\,(ux)$, $y = x^{-1}$, $u = \tau\,(vy)$;

or

b) the reduced word in F representing $w = x(\tau\,(ux))^{-1}\,vy$ has length at least two, begins with x, and ends with y.

iv) The words $ux\,(\tau\,(ux)^{-1})$, $u \in U$ and $x \in T$ with $ux \notin U$, are all distinct and the set of them is equal to $B \cup B^{-1}$ where

$B = \{ux\,(\tau\,(ux))^{-1} : u \in U, x \in S\} \setminus \{1\}$.

Proof:

i) $ux\,(\tau\,(ux))^{-1} = 1$ if and only if $ux = \tau\,(ux)$ if and only if $ux \in U$.

ii ) As $Hux = H\tau(ux)$,

$Hu = H\tau(ux)x^{-1} = H\tau(\tau(ux)x^{-1})$ and so $u = \tau(\tau(ux)x^{-1})$ since both are in U.

iii) The crux of the proof of the Nielsen-Schreier theorem is the following:

Let $\tau(ux) = r_1 \ldots r_m$, $v = t_1 \ldots t_n$ be reduced words (all $r_j, t_j \in T$) so that $w = x\, r_m^{-1} \ldots r_1^{-1}\, t_1 \ldots t_n\, y = x(\tau(ux))^{-1} vy$. Let us examine this word.

If $x\, r_m^{-1} = 1$, then $\tau(ux)\, x^{-1} = r_1 \ldots r_m\, x^{-1} = r_1 \ldots r_{m-1} \in U$ by Lemma 1.2.2, which implies $u = \tau(\tau(ux)x^{-1})$ by part ii), $= \tau(ux)\, x^{-1}$, which implies $ux = \tau(ux)$, which implies $ux \in U$ by part i), a contradiction. Also $t_n y = 1$ implies $vy = t_1 \ldots t_{n-1} \in U$ by Lemma 1.2.2, another contradiction.

Therefore $xr_m^{-1}$ and $t_n y$ are both reduced words. Let $(\tau(ux))^{-1} v$ be equal to the reduced word $r_m^{-1} \ldots r_{i+1}^{-1}\, t_{i+1} \ldots t_n$. We have four cases:

1) $i<m$ and $i < n$, then case b) holds.

2) $i = m < n$, w is the reduced word $x\, t_{m+1} \ldots t_n\, y$, and b) holds; or

$t_{m+2} \ldots t_n y$, and here $xt_{m+1} = 1$ which implies $\tau\ (ux)\, x^{-1}$

$= r_1 \ldots r_m x^{-1} = t_1 \ldots t_{m+1} \in U$ by Lemma 1.2.2, which implies

$u = \tau\ (\tau\ (ux)x^{-1}) = \tau\ (ux)x^{-1}$ by parts ii) and i), which implies

$ux = \tau\ (ux)$, which implies $ux \in U$, a contradiction.

3) $i = n < m$, w is the reduced word $xr_m^{-1} \ldots r_{n+1}^{-1}\, y$, and b) holds;

or $xr_m^{-1} \ldots r_{n+2}^{-1}$ and here $r_{n+1}^{-1}\, y = 1$ which implies

$vy = r_1 \ldots r_{n+1} \in U$ by Lemma 1.2.2, a contradiction.

4) $i = m = n$, $w = xy$, and b) holds; or $w = 1$ whereupon $y = x^{-1}$,

$\tau\ (ux) = v$ and so by part ii), $\tau\ (vy) = \tau\ (\tau\ (ux)x^{-1}) = u$.

Therefore b) holds in all cases except the last which yields a).

iv) Let $ux(\tau\ (ux))^{-1} = vy\ (\tau\ (vy))^{-1}$ where $ux, vy \in UT\backslash U$,

and then $x\ (\tau\ (ux))^{-1}\ (\tau\ (vy))\, y^{-1} = u^{-1} v.$ (4)

By parts i) and ii), $\tau\ (vy)\, y^{-1} \notin U$, since if $\tau\ (vy)\, y^{-1} \in U$, then

$\tau\ (\tau\ (vy)y^{-1}) = v = \tau\ (vy)\, y^{-1}$ which implies $vy = \tau\ (vy)$, a

contradiction since $vy \notin U$.

So we can apply part iii) to the left hand side of (4) which is equal to either

a) 1, where $u = v$, and $x = y$; or

b) a reduced word of the form $x \ldots y^{-1}$ so that $u^{-1}v \neq 1$ and reducing the right hand side, either u ends in $x^{-1}$ so that $ux \in U$, or v ends in $y^{-1}$ so $vy \in U$ (using Lemma 1.2.2), a contradiction. Therefore the words $ux(\tau(ux))$-1, $u \in U$, $x \in T$, $ux \notin U$, are all distinct.

Letting $B_1 = \{ux(\tau(ux))^{-1}: u \in U, x \in S^{-1}\} \setminus \{1\}$, the above implies the set in question is $B \cup B_1$. Since, for $u \in U$, $x \in T$, we have

$(ux(\tau(ux))^{-1})^{-1} = \tau(ux)x^{-1}u^{-1} = \tau(ux)x^{-1}(\tau(\tau(ux)x^{-1}))^{-1}$,

by part ii), we have $B^{-1}$ is a subset of $B_1$, and $B_1^{-1}$ is a subset of B, so that $B_1 = B^{-1}$ as required. ///

Step 5. The Main Theorem.

1.2.5 Theorem (Nielsen - Schreier) If F is free of rank r and H is a subgroup of F, then H is free. If $[F:H] = g$ is finite, then the rank of H is equal to $(r-1)g + 1$.

Proof:

We prove, using the above notation, that H is free on the set B. Since $A = B \cup B^{-1} \cup \{1\}$ (Lemma 1.2.4 part (iv)), and A generates H, (Lemma 1.2.3), we see that B generates H. Let $b_1 \ldots b_n$, $n \geq 1$ be a reduced word in the elements of $B \cup B^{-1} = A \setminus \{1\}$, say $b_i$ $= u_i x_i (\tau(u_i x_i))^{-1}$ for $1 \leq i \leq n$, where $u_i \in U$, $x_i \in T$, $u_i x_i \notin U$, and consider the product $b_i b_{i+1} = u_i x_i (\tau(u_i x_i))^{-1} u_{i+1} x_{i+1} (\tau(u_{i+1} x_{i+1}))^{-1}$ for some i between 1 and n.

Since $b_1 \ldots b_n$ is reduced, $b_i b_{i+1} \neq 1$, and so by Lemma 1.2.4 iii), $x_i (\tau(u_i x_i))^{-1} u_{i+1} x_{i+1}$ is equal to a reduced word in the elements of T of the form $x_i \ldots x_{i+1}$, of length at least two. For $x_i (\tau(u_i x_i))^{-1} u_{i+1} x_{i+1} = 1$ implies $u_{i+1} = \tau(u_i x_i)$, $x_{i+1} = x_i^{-1}$, $u_i = \tau(u_{i+1} x_{i+1})$, which implies $b_i b_{i+1} = 1$, a contradiction.

Therefore $b_1 \ldots b_n = \ldots x_1 \ldots x_2 \ldots \ldots \ldots x_n \ldots$

$= u_1 x_1 (\tau(u_1 x_1))^{-1} u_2 x_2 (\tau(u_2 x_2))^{-1} \ldots u_n x_n (\tau(u_n x_n))^{-1}$

$= u_1x_1 \ldots \; x_2 \ldots x_3 \; \ldots \ldots \ldots \; x_{n-1} \ldots x_n \, (\tau \, (u_nx_n))^{-1}$ the right hand side being a reduced word in T which has length at least $n \geq 1$, since $u_1$ does not end in $x_1^{-1}$, and$(\tau \, (u_nx_n))^{-1}$ does not begin with $x_n^{-1}$.

Therefore $b_1 \ldots b_n \neq 1$, and there is no nontrivial relation in H among the elements of B. Therefore H is free on B.

Note in the case that the index of H in F is finite we have a finite set of free generators for H.

Now we prove the numerical part of the theorem when H is a normal subgroup N of F. Let N have finite index k in F. Modify the construction of the Schreier transversal by restricting attention to reduced words $x_1 \ldots x_n \in F$ with each $x_i \in S$ (rather than $\in$ T). If such words are called *positive* we choose as representatives in U of any right coset to be the least positive word in that coset ( or 1, if the coset is N itself).

Everything said above goes through, and in particular the Schreier property holds, provided we can show that each nontrivial

coset does in fact have a positive element.

If $x \in T$, then $Nx \in F/N = G$ say, a group of order k, so that by Lagrange's theorem, $(Nx)^k = N$, that is $x^k \in N$. Let $x_1 \ldots x_n$ be any reduced word in T. Then for all i with $x_i^{-1}$ is S,

$Nx_1 \ldots x_n = Nx_1 \ldots x_{i-1} (Nx_i) x_{i+1} \ldots x_n$, since N is normal,

$= Nx_1 \ldots x_{i-1}(Nx_i^{-k}x_i) x_{i+1} \ldots x_n$, as $x_i^k \in N$,

$= Nx_1 \ldots x_{i-1} Nx_i^{-k+1} x_{i+1} \ldots x_n$

$= Nx_1 \ldots x_{i-1} x_i^{-k+1}x_{i+1} \ldots x_n$

and $x_i^{-k+1} = (x_i^{-1})^{k-1}$ is a reduced word in S. Performing this operation for each i, with $x_i^{-1} \in S$, we obtain $Nx_1 \ldots x_n = Nw$ where w is a reduced word in S (rather than $\in T$), so that $w \in Nx_1 \ldots x_n$ as required. Call the resulting transversal U. Consider the elements ux, $u \in U, x \in S$.

Since the $ux\ (\tau(ux))^{-1} \neq 1$ are all distinct by Lemma 1.2.4 iv), and there are kr of them all together, we must show that precisely

k-1 of them are 1, that is, precisely k-1 of the ux belong to U. If

$v = x_1 \ldots x_n \in U\backslash\{1\}$ (so that all $x_i \in S$, $n \geq 1$), then $v = ux_n$

with $u \in U$.

So every element of $U \setminus \{1\}$ appears in the set of ux's, that is,

$US \cap U = U\backslash\{1\}$, as required.

Note that $ux = u'x' \in U/\{1\}$ implies $u = u'$ and $x = x'$ since both ux and u'x' contain only elements of S and are thus reduced.

Let H a subgroup of F be arbitrary of finite index g. Let C be the set of all right cosets of H in F, and for each $w \in F$, let $\Phi_w$ be the mapping

$\Phi_w : C \rightarrow C$

$Hv \rightarrow Hvw.$

Each $\Phi_w$ is 1-1 and onto C.

We get a mapping

$\Phi : F \rightarrow S_g$

$w \rightarrow \Phi_w$

of F into the symmetric group of degree g, which is a homomorphism. The kernel of $\Phi$ is a normal subgroup of F contained in H and having index at most g!. Put N = Ker $\Phi$.

Then N is a normal subgroup of F, N is a subgroup of H, so N is a normal subgroup of H, and [F:N] is finite. Let [H:N] =h, so that [F:N] = [F:H][H:N] = gh. Since N is normal, r(H) = rank of H, r(N) = rank of N, N is a normal subgroup of F and [F:N] = gh, we have r(N) = (r-1)gh +1. Since N is a normal subgroup of H and [H:N] = h, we have r(N) = (r(H) -1)h + 1. But then (r-1)gh + 1 = (r(H) -1) h + 1, so r(H) = (r-1)g + 1 as required. ///

1.3 FREE PRESENTATIONS OF GROUPS

Suppose X is a set, F is the free group on X, R is a subset of F (and so consists of words in elements of X), N is the normal closure of R in F (sometimes denoted by [R]), and G is the factor group F/N.

1.3.1 Definition - With this notation, we write

$$G = \langle X \mid R \rangle \qquad (1)$$

and call the right hand side of this equation a *free presentation*, or simply a *presentation* of G. The elements of X are called *generators*

and those of R are called *relators*. A group G is called *finitely presented* if it has a presentation of the form (1) such that both X and R are finite sets.

1.3.2 Theorem - Every group has a presentation, and a finite group is finitely presented.

Proof:

Let G be a group, and let $\Phi$: F $\rightarrow$ G be an epimorphism with F the free group on the set G' underlying G. Then G = <G'| Ker $\Phi$ >. When G is finite, we repeat the process, but replace Ker $\Phi$ by a set R of free generators for Ker $\Phi$. Then G = <G'|R> and both |G'| and |R| are finite by Theorem 1.2.5. In fact if |G| = g, then |G'| = g, and |R'| = $g^2$-g+1. ///

1.3.3 Lemma - Let X, Y, Z be groups and let $\alpha$: X $\rightarrow$ Y , $\beta$ : X $\rightarrow$ Z be homomorphisms with $\alpha$ onto and such that Ker $\alpha$ is subgroup of Ker $\beta$. Then there exists a homomorphism $\gamma$ : Y $\rightarrow$ Z such that $\gamma \circ \alpha = \beta$

Proof:

For any y $\in$ Y, choose x $\in$ X such that $\alpha(x) = y$ and define $\gamma(y) = \beta(x)$. First we show this is well defined.

Let $x' \in X$ be another preimage of y, such that $\alpha(x') = y = \alpha(x)$. Then $x'x^{-1} \in \text{Ker } \alpha$, a subset of Ker $\beta$, so that $\beta(x'x^{-1}) = 1$, whence $\beta(x')$ $= \beta(x)$ ( $= \gamma(y)$). Therefore the definition of $\gamma(y)$ is independent of the choice of x. For any $x \in X$, x is the preimage of $\alpha(x)$ under $\alpha$, such that, since $\gamma(y) = \beta(x)$, $(\gamma \circ \alpha)(x) = \gamma(\alpha(x)) = \beta(x)$. We show that $\gamma$ is a homomorphism.

Let $y_1, y_2 \in Y$, and $x_1, x_2 \in X$ be their preimages. Then $\alpha(x_1x_2)$ $= \alpha(x_1)\ \alpha(x_2) = y_1\ y_2$, since $\alpha$ is a homomorphism, so that $x_1x_2$ is a preimage of $y_1y_2$. Then since $\gamma(y) = \beta(x)$ and $\beta$ is a homomorphism, $\gamma(y_1y_2) = \beta(x_1x_2) = \beta(x_1)\ \beta(x_2) = \gamma(y_1)\ \gamma(y_2)$, completing the proof . ///

1.3.4 Theorem. -- If R and S are subsets of the free group F on a set X such that R is a subset of S, then there is an epimorphism $\gamma : \langle X|R\rangle \rightarrow \langle X|S\rangle$ such that $\gamma(x[R]) = x[S]$ for $x \in X$.

Proof:

This is a simple application of the above lemma with $\alpha$ and $\beta$ the natural maps. Let $\alpha$: $F \rightarrow F/[R]$ and $\beta : F \rightarrow F/[S]$, be the natural

maps. Then $\alpha$ is onto, and Ker $\alpha$ is a subset of Ker $\beta$. Therefore by Lemma 1.3.3 there is a map $\gamma : F/[R] \rightarrow F/[S]$ such that $\gamma \circ \alpha = \beta$. Since $\beta$ is onto, so is $\gamma$. Also $\gamma(w[R]) = w[S]$ for $w \in F$. ///

1.3.5 Theorem - Let $G = \langle X|R \rangle$ be a group, with $X = \{x_1, \ldots, x_n\}$ and $R = \{r_1, \ldots, r_m\}$. Let H be a group and $\Phi : X \rightarrow H$ be a function such that $\Phi(x_i) = x_i'$ $1 \leq i \leq n$, say, and let w' be the word obtained by priming each of the letters in any word $w \in W(X)$. Suppose that each $r_i'$ is the identity of H. Then there is a mapping $\Phi'$: $G \rightarrow H$ such that $\Phi'(w) = w'$ is well defined and is a homomorphism from $G \rightarrow H$. In particular if the $x_i'$ generate H, then $\Phi'$ is onto, and H is a homomorphic image of G, so that $|H| \leq |G|$ ( in fact $|H|$ divides $|G|$ in the case when G is finite ).

Proof:

Let $F = \langle X \mid \rangle$ and let $\nu$ be the natural map from F to G where $\nu: F \rightarrow F/[R]$ and $[R] = \text{Ker } \nu$. Since F is free on X, $\Phi$ extends uniquely to a homomorphism $\Phi''$: $F \rightarrow H$. Our assumption about the $r_i'$

means that $R \subset \text{Ker } \Phi''$, and by the definition of normal closure, we have $[R] \subset \text{Ker } \Phi''$ also, that is $\text{Ker } \nu \subset \text{Ker } \Phi''$.

Then by Lemma 1.3.3 there is a homomorphism $\Phi'$: $G \rightarrow H$ such that $\Phi' \circ \nu = \Phi''$. If in addtion, the $x_i'$ generate H, then $\Phi''$ is onto, so $\Phi'$ is onto also. ///

## 1.4 TIETZE TRANSFORMATIONS

Given a presentation $G = \langle X \mid R \rangle$ a Tietze transformation $T_i$ $(1 \leq i \leq 4)$ transforms it into a presentation $\langle X' \mid R' \rangle$ in accordance with:

### 1.4.1 Definition

1. If r is a word in W(X) and $r = 1$ is a relation which holds in G, let $X' = X$, $R' = R \cup \{r\}$.

2. If $r \in R$ is such that the relation $r = 1$ holds in the group $\langle X|R\backslash\{r\}\rangle$ then let $X' = X$ and $R' = R\backslash\{r\}$.

3. If w is a word in W(X) and z is a symbol not in X, put $X' = X \cup \{z\}$, $R' = R \cup \{ w z^{-1}\}$.

4. Finally if $z \in X$ and w is a word in the elements of X other than z

such that $w z^{-1} \in R$, then take $X' = X \setminus \{z\}$ and substitute w for z in every other element of R to get R'.

1.4.2 Theorem. The application of any of the four Tietze transformations does not affect the isomorphism class of the group presented.

Proof:

For $T_1$ the identity map $\langle X| \ \rangle \rightarrow \langle X'| \ \rangle$ carries [R] onto [R']

and so induces an isomorphism $\langle X|R\rangle \rightarrow \langle X'| \ R'\rangle$ in this case. Note $F = \langle X| \ \rangle = F' = \langle X' \ | \ \rangle$, [R] = [R'], and so F/[R] is isomorphic to F'/[R']. The inverse of this isomorphism yields the result for $T_2$.

As for $T_3$, there is a homomorphism $\Phi$: $G \rightarrow \langle X' \ | \ R' \rangle$ fixing X by Theorem 1.3.5. By the same theorem, the mapping $X' \rightarrow G$ fixing X and sending z to w extends to a homomorhism $\Phi'$: $\langle X'|R'\rangle \rightarrow G$.

Note $\Phi'$ is the inverse of $\Phi$ and so $\langle X|R\rangle$ is isomorphic to $\langle X'|R'\rangle$. For

$\Phi \circ \Phi' (x [R \cup \{wz^{-1}\}]) = \Phi (x [R]) = x [R \cup \{w z^{-1}\}]$,

$\Phi \circ \Phi' (z [R \cup \{wz^{-1}\}]) = \Phi (w [R]) = w [R \cup \{w z^{-1}\}] = z [R \cup \{w z^{-1}\}]$, and

$\Phi' \circ \Phi$ (x [R]) = $\Phi'$( x [R U $\{w z^{-1}\}$]) = x [R].

As for $T_4$, the inverse of $\Phi$, which is $\Phi'$, coupled with transformations of type $T_1$ and $T_2$ gives the required isomorphism in the final case. We have

$<X \cup \{z\} \mid w z^{-1} = 1 , R_1 (X,z) = R_2 (X,z) = \ldots = R_k(X,z) = 1>$

applying $T_1$, is isomorphic to,

$< X \cup \{ z\} \mid w z^{-1} = 1, R_1 (X,z) = R_2 ( X,z) = \ldots = R_k(X,z) = 1,$

$R_1'(X,w) = R_2'(X,w) = \ldots R_k' (X,w) = 1>$

applying $T_2$, is isomorphic to,

$\{X \cup \{z\} \mid w z^{-1} = 1, R_1' (X,w) = R_2'(X,w) = \ldots = R_k'(X,w) = 1>$

applying $\Phi'$, is isomorphic to,

$<X \mid R_1' (X,w) = R_2'(X,w) = \ldots R_k'(X,w) = 1> = <X' \mid R'>$. ///

1.4.3 Theorem- Given two finite presentations <X|R> and <Y|S> for a group G, one can be transformed into the other by a means of a finite number of Tietze transformations.

Proof:

We shall deal with relations rather than relators and write $R = 1$ to denote the set of equations $r=1$ , $r \in R$. We write $X = X(Y)$, $Y = Y(X)$ for the equations expressing the elements of X as words in Y, and vice versa (possible since both X and Y generate G).

The method of progress from $\langle X|R\rangle$ to $\langle Y|S\rangle$ is indicated in the following scheme.

$G = \langle X \mid R(X) = 1\rangle$.

$T_3$ $Y = Y(X)$ : $\langle X,Y \mid R(X) = 1, Y = Y(X)\rangle$.

$T_1$ $S(Y) = 1$ , $X = X(Y)$

: $\langle X, Y| R(X) = 1, S(Y) = 1, Y = Y(X) , X=X(Y)\rangle$.

$T_4$ $X = X(Y)$ : $\langle Y|$ $R(X(Y)) = 1, S(Y) =1, Y = Y(X(Y))\rangle$.

$T_2$ $R(X(Y)) = 1, Y = Y(X(Y))$ : $\langle Y \mid S(Y) = 1\rangle$.

Step 2 is valid because the elements $Y \subset G$ satisfy $S(Y)= 1$.

Step 4 is valid because $S(Y) = 1$ actually defines G in terms of the elements of Y. ///

Remarks: In the process of the above proof, the Tietze

transformations respectively have been applied

$T_1$: |S| + |X|, $T_2$: |R| + |Y|, $T_3$: |Y| , $T_4$ : | X| times.

1.4.4 - Theorem Let G = <X| R(X) = 1> and let Y ⊂ G be another set of generators of G such that X = X(Y), say. Then G is isomorphic to <Y| R(X(Y)) = 1, Y = Y(X(Y))>.

Proof:

Using the notation of the proof of Theorem 1.4.3, we proceed by applying Tietze transformations.

G = <X| R(X) = 1>.

$T_3$ Y = Y(X) : < X, Y| R(X) = 1, Y = Y (X)>.

$T_1$ X = X(Y) : < X,Y | R(X) = 1, Y = Y(X), X = X(Y)>.

$T_4$ X = X(Y) : < Y| R(X(Y)) = 1, Y = Y (X(Y))>. ///

Remarks: In the process of the above proof, the Tietze transformations respectively have been applied

$T_1$: |X|, $T_3$: |Y| , $T_4$: | X| times.

# CHAPTER II

## COSET ENUMERATION

We are interested in determining the index of a subgroup of a finitely presented group. An algorithm to solve this problem was discovered by Coxeter and Todd in 1936. Since then, various versions of this algorithm have been given, along with a proof of the correctness of the algorithm. We give a formal statement of the problem, a discussion of the algorithm and a proof of its correctness, and examples of the application of the algorithm to various groups and subgroups.

It should be noted that the algorithm is mechanical enough to be implemented on a computer. In fact, several people have written computer implementations of it.

## 2.1 STATEMENT OF THE PROBLEM

Our problem may be stated informally as: Find the index of H in K, where K is generated by $y_1, y_2, \ldots, y_n$ subject to the relations $r_1=r_2 = \ldots = r_s = 1$, and H is generated by $h_1, h_2, \ldots, h_t$. Here the $r_i$ and $h_i$ are expressed in terms of the $y_i$.

A formal restatement of the problem is: Let F be a free group on the generators $x_1, \ldots, x_n$. Let $r_1, \ldots, r_s$ be words in F, let R be the set of all their conjugates in F, and let [R] be the subgroup generated by R. Define K as F/[R] and $y_i$ as the image of $x_i$ under the canonical map. Furthermore, let $g_1, \ldots, g_t$ be elements of F, $h_1, \ldots, h_t$ their images in K, and H the subgroup generated by the $h_i$. It is required to find the index of H in K.

Let G be the subgroup of F generated by $g_1, \ldots, g_t$ and R. Under the canonical map, the cosets of H in K correspond to the cosets of G in F, and the original problem is equivalent to finding the index of G in F.

## 2.2 THE ALGORITHM

We will describe the algorithm to solve the coset enumeration problem in terms of operations on an array A of integers with n+1 columns and a varying number of rows. (It is possible that the number of rows may be infinite.)

The columns are labelled from 0 to n, with columns 1 to n corresponding to the n generators $x_1, \ldots, x_n$ of F and column 0 corresponding to the identity element which we will denote $x_0$. For notational convenience, we number the rows consecutively from 1, but the order of the rows has no significance. The array need not be complete, that is, some cells $A_{ij}$ may be empty, but we assume that every row contains at least one non-empty cell.

The following conditions are assumed to hold for all arrays:

1) Any row with more than one entry has an entry in column 0.

2) The integer 1 appears in the array.

3) Any integer appearing in the array appears at least once in each column.

4) No proper subset of the rows forms an array satisfying 3).

Two further properties are given for which it is convenient to have names.

We call an array *consistent* if

5) No integer appears more than once in any column.

We call an array *complete* if

6) No cell is empty.

An example of a complete and consistent array corresponding to a free group F with two generators x and y is as follows:

| 1 | x | y |
|---|---|---|
| 1 | 1 | 2 |
| 2 | 3 | 3 |
| 3 | 4 | 1 |
| 4 | 5 | 5 |
| 5 | 2 | 6 |
| 6 | 6 | 4. |

An array which is both consistent and complete can be interpreted as a multiplication table defining a transitive representation of F by permutations of the integers appearing in the array. For an integer p, $p \cdot x_j$ is defined as the entry in column j of the row with p in column 0, while $p \cdot x_j^{-1}$ is the entry in column 0 of the

row with p in column j.

Since F is free this specification of the action of the generators can be extended to give a representation of the whole group. Conditions 3), 5), 6) ensure that the operations are well defined.

A little effort is required to show condition 4) is equivalent to transitivity of the representation. For if the action were not transitive, we could delete all rows which begin with an integer which is not in the orbit of 1, violating condition 4). And if we were able to violate condition 4), certain integers would not be in the orbit of 1, which violates transitivity.

Given a complete, consistent array A, we define S'(A) to be the subgroup of F which leaves the integer 1 invariant, i.e. the isotropy subgroup of 1 under the action. The right cosets of S'(A) in F are in one to one correspondence with the different integers appearing in A, and so the index of S'(A) in F is equal to the (possibly infinite) number of rows of A. For example we could identify with the integer i the right coset $\{w \in F: 1 \cdot w = i\}$, where $\{w \in F: 1 \cdot w = 1\} = S'(A)$.

An array satisfying 1) - 4) can be interpreted as partially

specifying a representation on equivalence classes of the integers in A and defines a corresponding subgroup S(A). We explain as follows.

A *link* in A is a pair (i,u) where i is a row number and $u = x_j$ or $x_j^{-1}$, and neither $A_{i0}$ nor $A_{ij}$ is empty. The head of (i,u) is $A_{i0}$ if $u = x_j$ and $A_{ij}$ if $u = x_j^{-1}$, while the tail of (i,u) is $A_{ij}$ if $u = x_j$ and $A_{i0}$ if $u = x_j^{-1}$. A *sequence of links* $c = (i_1,u_1), \ldots, (i_m,u_m)$ is a *path* in A if the head of each link after the first is equal to the tail of the preceding link. We say that c is a path from p to q if p is the head of $(i_1,u_1)$ and q is the tail of $(i_m,u_m)$. The inverse of c is the sequence $(i_m,u_m^{-1}), \ldots, (i_1,u_1^{-1})$ and is a path if c is. The element of F represented by the word $u_1 \ldots u_m$ is said to be *covered by c.* Note that if c covers w then $c^{-1}$ covers $w^{-1}$.

We define S(A) as the set of all $w \in F$ covered by some path from 1 to 1 in A.

Note that S(A) is a subgroup of F. If $c_1$ is a path from 1 to 1 covering $w_1$, and $c_2$ is a path from 1 to 1 covering $w_2$, then $c_1c_2$ is a

path from 1 to 1 covering $w_1w_2$. Also $c_1^{-1}$ is a path from 1 to 1 covering $w_1^{-1}$. Since 1 appears somewhere in the first column of any array A, we have a path from 1 to 1 covering the identity. We define p and q in A to be equivalent if there is a path from p to q in A covering the identity. Then the distinct equivalence classes of integers correspond to distinct right cosets G/S(A). For example we could identify with the integer i the right coset $\{w \in F: \exists$ path from 1 to i covering w$\}$, where $\{w \in F: \exists$ path from 1 to 1 covering w$\}$ = S(A).

We next describe the effect that certain operations on A have on S(A).

2.2.1 Lemma- Suppose that the cell $A_{ij}$ is empty and that p is an integer not occurring in A. Form A' by placing p in $A_{ij}$ and for each column except the jth adding a new row with p in that column, so that condition 3) is satisfied. Then S(A') = S(A).

Proof:

Suppose $j \neq 0$. The only essential difference between A' and A is that A' contains additional links $(i,x_j)$ and $(i,x_j^{-1})$. Clearly S(A)

is a subset of S(A'). Conversely, take any $w \in S(A')$, and let c be a path from 1 to 1 in A' covering w and of minimal length. If either of the new links occurs in c, the combination $(i,x_j)(i,x_j^{-1})$ must occur, since $(i,x_j)$ is the only link in A' with p as tail and $(i,x_j^{-1})$ is the only one with p as head. An occurrence of $(i,x_j)(i,x_j^{-1})$ would contradict the minimality of c. Hence c is a path in A and $w \in S(A)$. If $j=0$, then by condition 1) there is a unique $k\neq 0$ with $A_{ik}$ non-empty and a similar consideration of $(i,x_k^{-1})$ and $(i,x_k)$ completes the proof. ///

2.2.2 - Lemma - Let $p<q$ be two integers in A and let $c_p$ and $c_q$ be paths in A from 1 to p and 1 to q respectively. Form A' from A by replacing all occurrences of q by p. Then S(A') is the subgroup generated by S(A) and $w_p w_q^{-1}$ where $w_p$, $w_q$ are covered by $c_p$,$c_q$ respectively.

Proof:

Let G be the subgroup generated by S(A) and $w_p w_q^{-1}$. Any sequence of links which is a path in A is also a path in A' and so

$S(A) \subset S(A')$. In A', $c_p c_q^{-1}$ is a path from 1 to 1 covering $w_p w_q^{-1}$. Hence $w_p w_q^{-1} \in S(A')$ and $G \subset S(A')$.

Conversely, suppose $w \in S(A')$ and $c = (i_1,u_1) \ldots (i_m,u_m)$ is a path from 1 to 1 in A' which covers w. Considered as a sequence of links in A, c can fail to be a path in A only because of breaks where the tail of one link is p and the head of the other link is q, or vice versa. Let b(c) denote the number of such breaks. (If p = 1, then c may not be from 1 to 1 in A. A trivial variation of the following argument will handle this possibility; we consider this case at the end of the following argument.) Assume the inductive hypothesis that if w is any element of S(A') covered by a path c from 1 to 1 in A' such that $b(c) < s$, then $w \in G$. The hypothesis is true for s=1, since if b(c) = 0, then c is a path in A and $w \in S(A)$ which is a subset of G. Now consider a $w \in S(A')$ covered by a path with b(c) = s. Suppose the first break occurs after the kth link, and for definiteness, suppose that the tail of $(i_k,u_k)$ is p, and that the head of $(i_{k+1},u_{k+1})$ is q. Consider the sequence $c'=s_1s_2s_3$ where $s_1 = (i_1,u_1) \ldots (i_k,u_k)c_p^{-1}$, $s_2 = c_p c_q^{-1}$,

and $s_3 = c_q(i_{k+1},u_{k+1}) \ldots (i_m,u_m)$. Now c' covers w, so $w = w_1w_2w_3$ where $w_i$ is covered by $s_i$. The sequence $s_1$ is a path from 1 to 1 in A and so $w_1 \in S(A) \subset G$; $w_2 = w_p w_q^{-1} \in G$; and $s_3$ is a path from 1 to 1 in A' with $b(s_3) < s$ so by the inductive hypothesis $w_3 \in G$. Hence $w \in G$ and the proof is complete.

Note if $p = 1$, and c is not from 1 to 1 in A, it may be a path from q to 1 in A, so we can consider $c'=s_1s_2s_3$ where $s_1 = c_p^{-1}$, $s_2 = c_pc_q^{-1}$, $s_3 = c_q(i_1,u_1)(i_2,u_2) \ldots (i_m,u_m)$, and apply the above argument to $s_3$ which is from 1 to 1 in A. Other cases are handled similarly. ///

2.2.3 Corollary - If p and q are equivalent in A, and A' is obtained by replacing q by p throughout A, then S(A')=S(A).

Proof:

Let $c_{pq}$ be a path from p to q covering the identity. Apply Lemma 2.2.2 with $c_p$ any path from 1 to p covering $w_p$ and take $c_q = c_pc_{pq}$. Since $w_p = w_q$, $w_pw_q^{-1}$ is the identity and the result

follows. ///

2.2.4 Lemma - Given a finite array A, there is a finite algorithm for obtaining a consistent array B with S(A) = S(B).

Proof:

Suppose A is not consistent so that, say $A_{ij} = A_{kj}$ with $i \neq k$. If either of the rows has only a single entry, it may be deleted without affecting S(A); otherwise both $A_{i0}$ and $A_{k0}$ are non-empty. For each column number $m \neq j$ proceed from left to right as follows:

i) If $A_{im} = A_{km}$, or if both are empty, make no change.

ii) If just one of $A_{im}$, $A_{km}$ is empty, copy the other entry into it.

iii) If $A_{im} \neq A_{km}$, replace the larger number by the smaller throughout the array.

Operation ii) does not affect S(A), since it does not create any essentially new paths, that is, paths that cover new words.

If iii) applies, then $A_{im}$ is equivalent to $A_{km}$, and so by Corollary 2.2.3, S(A) is unchanged.

When these opertions have been done for all columns the two

rows i and k are identical, and one may be deleted. Thus from any inconsistent array A we obtain an equivalent array A' with one less row. If A is finite the process must lead to a consistent array in a finite number of steps. ///

Remark: In a consistent array, no two distinct integers are equivalent. For if they were equivalent, then at least one integer would appear more than once in a column.

2.2.5 Lemma - Given an array A and a word $w \in F$, an array B may be constructed in a finite number of steps such that S(A) = S(B) and for some q there is a path from 1 to q in B which covers w.

Proof:

The construction can be made by applying the operation in Lemma 2.2.1 not more than m times, where m is the length of w. Note, w may not be reduced, so we consider the length of w to be the length of the non-reduced word w. ///

2.2.6 Theorem - Given a finite set of words $u_1, \ldots, u_m$ in a finitely generated free group, there is a finite algorithm for determining the index of the subgroup they generate.

Proof:

Let $G_k$, k = 1, . . ., m be the subgroup generated by the first k of the $u_i$ and let $G_0$ be the trivial subgroup. Start with the array $A_0$ which has n+1 rows, a 1 in each row and column (so there are n+1 entries in the array), and no other entries. Then $S(A_0) = G_0$. Suppose $A_{k-1}$ has been constructed, with $S(A_{k-1}) = G_{k-1}$. Apply Lemma 2.2.5 if necessary to obtain an array with a path covering $u_k$ from 1 to $p_k$, and then replace $p_k$ by 1 to obtain $A_k$. By Lemma 2.2.2, $S(A_k) = G_k$. Finally apply Lemma 2.2.4 to obtain a consistent array B with $S(B) = G_m$.

If B is complete, $G_m$ has finite index equal to the number of rows of B. On the other hand, if B is not complete, the construction of Lemma 2.2.1 leads to an array B' with $S(B') = G_m$. Examination of the construction shows that B' is also consistent and incomplete (unless n=1, B' actually has more empty cells than B) so the same operation can be applied to B', and so on. Carrying out this process countably

many times gives a complete consistent array, showing that $G_m$ has infinite index. ///

We are now in a position to attack the original problem. Let G be the subgroup of F generated by the words $g_1, \ldots, g_s$ and all conjugates of the words $r_1, \ldots, r_t$. Since the latter set is infinite we cannot simply apply Theorem 2.2.6. We describe a process for constructing a sequence of consistent arrays $A_0, A_1, \ldots$ which may or may not terminate. We write $G_k$ for $S(A_k)$.

Use the method of Theorem 2.2.6 to construct $A_0$ so that $G_0$ is generated by $g_1, \ldots, g_s$, and then continue as follows:

a) Take p as the smallest unprocessed integer in $A_k$. (Initially all integers are unprocessed and p=1 will be the choice when k=0.)

b) Let c be a path from 1 to p, and let w be the word it covers. Obtain a new consistent array $B_{k+1}$ by the method of Theorem 2.2.6 so that $S(B_{k+1})$ is generated by $G_k$ and the elements $wr_1w^{-1}, \ldots, wr_tw^{-1}$. Whenever a new integer is introduced in the construction use an

integer larger than any that has yet appeared; that is, an integer eliminated by use of Lemma 2.2.2 should not be reintroduced.

c) If necessary, apply Lemma 2.2.1 to fill all the cells of the row with p in column 0, and column 0 of all rows with p in other columns. Take the resulting array as $A_{k+1}$. Then $G_{k+1} = S(A_{k+1}) = S(B_{k+1})$ is generated by $G_k$ and $wr_1w^{-1}, \ldots, wr_tw^{-1}$.

d) Mark the integer p as processed, and increase k by 1.

e) If the array contains any unprocessed integers, return to step a). If it does not, stop.

The contention is that this procedure terminates if and only if G has finite index in F, and that if it does terminate then $G = S(A_f)$ where $A_f$ is the final array. If the process terminates, then, in virtue of step c) , $A_f$ must be complete and hence $G_f$ has finite index. Since $G_f$ is a subset of G, this shows that the process cannot terminate unless G has finite index.

Conversely, suppose G does have finite index. By Theorem 1.2.5, the Nielsen-Schreier theorem proved above in Chapter I, G is

finitely generated. Each member of a finite set of generators is expressible in terms of the g's and a finite number of the conjugates of the r's; hence G is actually generated by the g's and some finite number of conjugates of the r's. Now observe that if w is any word covered by a path from 1 to a processed integer in $A_k$, then all the conjugates $wr_iw^{-1}$ are in $G_k$. Because of step c), for any $w \in F$ there will be for some k a path in $A_k$ over w from 1 to some integer q. Eventually either q itself will be marked, or it will have been replaced by some smaller integer which was marked. Thus any finite set of conjugates of the r's will be in $G_k$ for some sufficiently large k and consequently, for some k, $G_k = G$ and is of finite index.

This implies $A_k$ will be complete, for if $A_k$ were not complete, $G_k = G$ would not have finite index by the argument given in the proof of Theorem 2.2.6. None of the steps a) - e) can introduce new integers into a complete array, so in a finite number of steps the algorithm will terminate. At this point $A_f = A_k$ and $S(A_f) = G_k = G$. Thus we

have proved:

2.2.7 Theorem - Given words $g_1, \ldots, g_s$ and $r_1, \ldots, r_t$ in a finitely generated free group, let G be the group generated by the g's and the conjugates of the r's. Then the above algorithm terminates if and only if G is of finite index in F. If the algorithm terminates, it determines the index of G in F. ///

2.3 EXAMPLES

The following examples will provide insight into the above algorithm.

When creating a path from 1 to 1 covering $w = u_1 u_2 \ldots u_k$, where $u_j \in X \cup X^{-1}$, $X = \{x_1, \ldots, x_n\}$, the notation will be to write $1\ u_1\ j_1\ u_2\ j_2 \ldots j_{k-1}\ u_k\ 1$ where $j_i$ are positive integers in the array for $i = 1$ to $k-1$. This is interpreted as a path from 1 to $j_1$ via $u_1$, a path from $j_1$ to $j_2$ via $u_2$, . . ., a path from $j_{k-1}$ to 1 via $u_k$.

Also note when processing the integer i, we need only create paths, using Lemma 2.2.1, from i to i covering the relator $r_j$. The reason being that if w is a word covered by a path from 1 to i, then

$w^{-1}$ is a word covered by a path from i to 1, and so creating a path from i to i covering the relator $r_j$ really adds the word $w\ r_j\ w^{-1}$ to the generators of S(A), which is what was intended. A path from i to i covering the relator $r_j$ is called an *$r_j$-cycle*.

When we create paths which cover words in the coset enumeration, we either make *deductions* or *definitions*. A *definition* is made when no information gives the product of a coset with a generator, so that a new integer must be defined in the array according to Lemma 2.2.1. A *deduction* at j is made when we create a cycle from i to i say, $i\ w_1\ i_1\ u_1\ j\ u_2\ i_2\ w_2\ i$, where i, j, $i_1$, and $i_2$ are positive integers in the array, $u_1, u_2 \in X \cup X^{-1}$, and $w_1$, $w_2 \in W(X)$. If we know $i_1\ u_1 = j$, but not $j\ u_2 = i_2$, then $j\ u_2 = i_2$ was deduced. Similarly, if we know $i_2\ u_2^{-1} = j$, but not $j\ u_1^{-1} = i_1$, then $j\ u_1^{-1} = i_1$ was deduced. Each cycle covered gives rise to 0 or 1 deductions, the deductions being made when we close the cycle.

When covering a path from i to i, we can make deductions from

left to right as well as right to left starting at i on either end. This reduces the number of new integers defined in the coset enumeration. When we come to a point where no deductions can be made, at this point only do we define new integers.

2.3.1 Let $G = \langle x,y | x^4 = y^3 = (xy)^2 = 1 \rangle$, $H = \langle x \rangle$.

Find the index of H in G.

| 1 | x | y | |
|---|---|---|---|
| 1 | | | Start with the identity. |
| | 1 | | 1x1. Add the generator for H. |
| | | 1 | Process integers starting with 1. |

| 1 | x | y | |
|---|---|---|---|
| 1 | 1 | 2 | 1x1x1x1x1. |
| | | | 1y2y3y1. |
| 2 | 3 | 3 | 1x1y2x3y1. |
| | 2 | | |
| 3 | | 1 | |

| 1 | x | y | |
|---|---|---|---|
| 1 | 1 | 2 | 2x3x4x5x2. |
| | | | 2y3y1y2. |
| 2 | 3 | 3 | 2x3y1x1y2. |
| 3 | 4 | 1 | |
| 4 | 5 | | |
| | | 4 | |
| 5 | 2 | | |
| | | 5 | |

| 1 | x | y | |
|---|---|---|---|
| 1 | 1 | 2 | |
| 2 | 3 | 3 | 3x4x5x2x3.<br>3y1y2y3. |
| 3 | 4 | 1 | 3x4y5x2y3. |
| 4 | 5 | 5 | |
| | | 4 | |
| 5 | 2 | | |

| 1 | x | y | |
|---|---|---|---|
| 1 | 1 | 2 | |
| 2 | 3 | 3 | 4x5x2x3x4.<br>4y5y6y4. |
| 3 | 4 | 1 | 4x5y6x6y4. |
| 4 | 5 | 5 | |
| 5 | 2 | 6 | |
| 6 | 6 | 4 | |

The array is complete so we are done. Note then that the index of H in G is 6.

2.3.2 Let $G = \langle x,y \mid x^3 = y^3 = 1 \rangle$, $H = \langle xy, yx \rangle$.

Find the index of H in G.

| 1 | x | y | |
|---|---|---|---|
| 1 | | | Start with the identity. |
| | 1 | | Add the generators for H. 1x2y1.<br>1y3x1. Then process integers |
| | | 1 | starting with 1. |

| 1 | x | y | |
|---|---|---|---|
| 1 | 2 | 3 | |
| 2 | 3 | 1 | 1x2x3x1. |
| | | | 1y3y2y1. |
| 3 | 1 | 2 | |

The array is complete and we are done. The index of H in G is 3.

So as to make as many deductions as possible without defining new integers in the array, sometimes it is desirable to check other relators and partially process other integers before an integer has been completely processed. This does not affect the termination of the algorithm, so long as eventually every integer is completely processed. This method will be illustrated in the following example.

2.3.3 Let $G = \langle a,b,c,d,e \mid abc^{-1} = bcd^{-1} = cde^{-1} = dea^{-1} = eab^{-1} = 1 \rangle$.

Let $H = \langle a \rangle$. Find the index of H in G.

| 1 | a | b | c | d | e | |
|---|---|---|---|---|---|---|
| 1 | | | | | | |
| | 1 | | | | | Start with the identity. |
| | | 1 | | | | Add the generator for H. |
| | | | 1 | | | 1a1. |
| | | | | 1 | | Process integers starting with 1. |
| | | | | | 1 | |

| 1 | a | b | c | d | e | |
|---|---|---|---|---|---|---|
| 1 | 1 | 2 | 2 | 3 | 3 | $1a1b2c^{-1}1.$ |
| | 1 | | | | | $1b2c3d^{-1}1.$ |
| | | | | | | $1d3e1a^{-1}1.$ |
| | | | 1 | | | $3e1a1b^{-1}3.$ |
| | | | | 1 | | $3b1c2d^{-1}3.$ |
| | | | | | | $3c3d2e^{-1}2.$ |
| 2 | | | 3 | 3 | 2 | $3d2e2a^{-1}3.$ |
| | | | | | | $1e3a2b^{-1}1.$ |
| | | | | | | $1c2d3e^{-1}1.$ |
| 3 | 2 | 1 | | 2 | 1 | |
| | 3 | | | | | |
| | | 3 | | | | |

At this point we have a 1d3 and 2d3 which makes 1 equivalent to 2.

Now replace 2 by 1 to make the array consistent.

| 1 | a | b | c | d | e |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 3 | 3 |
| | 1 | | | | |
| | | | 1 | | |
| | | | | 1 | |
| 1 | | | 3 | 3 | 1 |
| 3 | 1 | 1 | | 1 | 1 |
| | 3 | | | | |
| | | 3 | | | |

Now we have 1a1 and 3a1 which makes 1 and 3 equivalent.

So replace 3 by 1 and we are done.

| 1 | a | b | c | d | e |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 |

Therefore the index of H in G is 1, and so G is cyclic.

2.3.4 Let $G = \langle x^3=y^3 = (xy)^2 = 1 \rangle$ , $H = \langle x \rangle$.

Find the index of H in G.

| 1 | x | y | |
|---|---|---|---|
| 1 | | | Start with the identity. |
| | 1 | | Add the generator for H. |
| | | 1 | 1x1. |
| | | | Process integers starting with 1. |

| 1 | x | y | |
|---|---|---|---|
| 1 | 1 | 2 | 1x1x1x1. |
| | | | 1y2y3y1. |
| 2 | 3 | 3 | 1x1y2x3y1. |
| | 2 | | |
| 3 | | 1 | |

| 1 | x | y | |
|---|---|---|---|
| 1 | 1 | 2 | 2x3x4x2. |
| 2 | 3 | 3 | 2y3y1y2. |
| | | | 2x3y1x1y2. |
| 3 | 4 | 1 | |
| 4 | 2 | | |
| | | 4 | |

| 1 | x | y | |
|---|---|---|---|
| 1 | 1 | 2 | 3x4x2x3. |
| 2 | 3 | 3 | 3y1y2y3. |
| | | | 3x4y4x2y3. |
| 3 | 4 | 1 | |
| 4 | 2 | 4 | |

At this point the array is complete and we are done. The index of H in G is 4.

2.3.5 Let $G = \langle x,y \mid x^2 = y^4 = xyx^{-1}y = 1 \rangle$, $H = \langle xy \rangle$.

Find the index of H in G.

| 1 | x | y | |
|---|---|---|---|
| 1 | | | Start with the identity. |
| | 1 | | Add the generator of H. |
| | | 1 | 1x2y1. |
| | | | Process integers starting with 1. |

| 1 | x | y | |
|---|---|---|---|
| 1 | 2 | 3 | |
| | | | 1x2x1. |
| 2 | 1 | 1 | 1y3y4y2y1. |
| | | | $1x2y1x^{-1}2y1$. |
| 3 | | 4 | |
| | 3 | | |
| 4 | | 2 | |
| | 4 | | |

| 1 | x | y | |
|---|---|---|---|
| 1 | 2 | 3 | 2x1x2. |
| 2 | 1 | 1 | 2y1y3y4y2. |
| 3 | | 4 | $2x1y3x^{-1}4y2.$ |
| 4 | 3 | 2 | |
| | 4 | | |

| 1 | x | y | |
|---|---|---|---|
| 1 | 2 | 3 | 3x4x3. |
| 2 | 1 | 1 | |
| 3 | 4 | 4 | |
| 4 | 3 | 2 | |

The array is complete and we are done. Therefore the index of H in G is 4.

## 2.4 ENHANCED COSET ENUMERATION (LEECH)

### 2.4.1 The Basic Method

Given an element of a group, presented as a word in the generators of the group, and an enumeration of the cosets of a subgroup of the group, we can readily determine whether the element is an element of the subgroup. All we have to do is to begin with coset 1 -- which is the subgroup itself -- and multiply by the successive letters of the word; the element is an element of the subgroup if and only if the final result is coset 1. Although this will exhibit which elements of the group are elements of the subgroup, it

does not enable us to exhibit them as words in the subgroup generators. The present method was devised to allow derivation of the appropriate words in the subgroup generators.

Recall that a cycle over a relator $r_j$ is a path in an array A from i to i covering a relator $r_j$ for some i and j positive integers.

Also recall that when we create paths which cover words in the coset enumeration, we either make deductions or definitions. A definition is made when no information gives the product of a coset with a generator, so that a new integer must be defined in the array according to Lemma 2.2.1. A deduction at j is made when we create a cycle from i to i say, i $w_1$ $i_1$ $u_1$ j $u_2$ $i_2$ $w_2$ i, where i, j, $i_1$, and $i_2$ are positive integers in the array, $u_1, u_2 \in X \cup X^{-1}$, and $w_1, w_2 \in W(X)$. If we know $i_1 u_1 = j$, but not $j u_2 = i_2$, then $j u_2 = i_2$ was deduced. Similarly, if we know $i_2 u_2^{-1} = j$, but not $j u_1^{-1} = i_1$, then $j u_1^{-1} = i_1$ was deduced. Each cycle covered gives rise to 0 or 1 deductions, the deductions being made when we close the cycle.

The enhanced coset enumeration handles deductions by

creating *deduction words.* Suppose i $w_1$ $i_2$ $u_1$ j $u_2$ $i_2$ $w_2$ i is a cycle from i to i, and we have deduced j $u_2$ = $i_2$. Then to get the deduction word corresponding to the above, we create a path from j to $i_2$ without using $u_2$, that is, j $u_1^{-1}$ $i_1$ $w_1^{-1}$ i $w_2^{-1}$ $i_2$. The *deduction word* would be $u_1^{-1}$ $w_1^{-1}$ $w_2^{-1}$, the word covered by the path from j to $i_2$. We will use the deduction word as a substitution when trying to write words in the subgroup in terms of the generators of the subgroup. This method will be illustrated in the following example.

2.4.2 Example.

Consider G = <a, b | $a^4 = b^3 = (ab)^2 = 1$>, H = <a>.

Find the index of H in G using enhanced coset enumeration.

| 1 | a | b | |
|---|---|---|---|
| 1 | | | Start with the identity. |
| | 1 | | Add the generator for H, |
| | | 1 | 1a1. |

Then process integers. We have 1a1a1a1a1. Next we have 1b2b3b1. So we defined 1b2, and 2b3. We deduced 3b1. So deduction #1 is 3b1 and 3b1 = $3b^{-1}2b^{-1}1$, a path from 3 to 1 not using

3b1.

Continuing, we have1a1b2a3b1. We have deduced 2a3. Deduction #2 is $2a3 = 2b^{-1}1a^{-1}1b^{-1}3$. We must process the deduction completely. We create deduction words recursively. Therefore, if we use a previous deduction path in creating a new deduction path, we substitute the deduction word for that path in the new path at the appropriate place. So deduction # 2 is $2a3 = 2b^{-1}1a^{-1}1b^{-1}3 = 2b^{-1}1a^{-1}1b2b3$ from above using the deduction $1b^{-1}3 = 1b2b3$.

Next we have 2a3a4a5a2. We defined 3a4 and 4a5. We deduced 5a2. Deduction #3 is $5a2 = 5a^{-1}4a^{-1}3a^{-1}2$ $= 5a^{-1}4a^{-1}3b^{-1}2b^{-1}1a1b2$.

As we continue processing integers, we have 2b3b1b2, 2a3b1a1b2, 3a4a5a2a3, 3b1b2b3, and 3a4b5a2b3. We have deduced 4b5. Deduction #4 is $4b5 = 4a^{-1}3b^{-1}2a^{-1}5$ $= 4a^{-1}3b^{-1}2b^{-1}1a^{-1}1b2b3a4a5$.

As we continue processing integers, we have 4a5a2a3a4, and 4b5b6b4. We have defined 5b6, and we have deduced 6b4. Deduction

#5 is $6b4 = 6b^{-1}5b^{-1}4 = 6b^{-1}5a^{-1}4a^{-1}3b^{-1}2b^{-1}1a1b2b3a4$.

As we continue processing integers, we have 4a5b6a6b4. We have deduced 6a6. Deduction #6 is $6a6 = 6b^{-1}5a^{-1}4b^{-1}6$ $= 6b^{-1}5a^{-1}4a^{-1}3b^{-1}2b^{-1}1a^{-1}1b2b3a4a5b6$.

At this point the array is complete. We have:

| 1 | a | b |
|---|---|---|
| 1 | 1 | 2 |
| 2 | $3_2$ | 3 |
| 3 | 4 | $1_1$ |
| 4 | 5 | $5_4$ |
| 5 | $2_3$ | 6 |
| 6 | $6_6$ | $4_5$ |

The subscripts correspond to the above deductions.

Now we have much more information in our array which can be demonstrated as follows. The deductions are

#1 $3b1 = b^{-1}b^{-1}$,

#2 $2a3 = b^{-1}a^{-1}bb$,

#3 $5a2 = a^{-1}a^{-1}b^{-1}b^{-1}ab$,

#4 $4b5 = a^{-1}b^{-1}b^{-1}a^{-1}bbaa$,

#5 $6b4 = b^{-1}a^{-1}a^{-1}b^{-1}b^{-1}abba$, and

#6 6a6 = $b^{-1}a^{-1}a\text{-}1b^{-1}b^{-1}a^{-1}bbaab$.

Now suppose we want to find the order of $c = a^{-1}b$. Then $c^4 \in H$ since $1c^41$, that is we have a path in the array from 1 to 1 covering $c^4$. $1a^{-1}1b2a^{-1}5b6a^{-1}6b4a^{-1}3b1$. Now if we substitute our deduction words in this word, we can express $c^4$ in terms of the generators of H. We have

$(1a^{-1}1b2)(2a^{-1}5)(5b6)(6a^{-1}6)(6b4)(4a^{-1}3)(3b1)$

$= (a^{-1}b)(b^{-1}a^{-1}bbaa)(b)(b^{-1}a^{-1}a^{-1}b^{-1}b^{-1}abbaab)$

$$(b^{-1}a^{-1}a^{-1}b^{-1}b^{-1}abba)(a^{-1})(b^{-1}b^{-1}) = 1,$$

since all the terms cancel in the above expression. Therefore in terms of the generators of H, $c^4 = 1$, so the order of c is 4.

Using the above method, one can also verify that the period of $d = a^{-1}b^{-1}ab$ is 3. One may obtain $d^3 = a^{-4}$, expressed in terms of the generators of H, but reference to the relator $a^4 = 1$ completes the verification. Therefore, when an element is expressed as a word of the subgroup, it is not necessarily derived in its simplest form.

Note that this method works because we are always making substitutions that are validly defined from the relators. For instance,

from $b^3=1$, allows us to substitute $b = b^{-2}$. Therefore 3b1 $= 3b^{-1}2b^{-1}1$ is a valid substitution.

Furthermore, consider the array B obtained from the final array A, by deleting the entries with subscripts corresponding to deduction words. Whenever we have a path from 1 to 1 in A, we are able to substitute equivalent deduction words which allows us to write the path from 1 to 1 in B. But the only paths from 1 to 1 in B are expressed in terms of the generators for H, namely S(B) = <a>. Therefore for every path from 1 to 1 in A, we are able to express it in terms of an equivalent word coverered by 1 to 1 in B, which is expressible in terms of <a> alone.

| 1 | a | b |
|---|---|---|
| 1 | 1 | 2 |
| 2 | | 3 |
| 3 | 4 | |
| 4 | 5 | |
| 5 | | 6 |
| 6 | | |

We can make the above array a valid array B by adding the appropriate entries so that each number appearing in the array appears at least once in each column. That is:

| 1 | a | b |
|---|---|---|
| 1 | 1 | 2 |
| 2 | | 3 |
| 3 | 4 | |
| 4 | 5 | |
| 5 | | 6 |
| 6 | | |
| | | 1 |
| | 2 | |
| | 3 | |
| | | 4 |
| | | 5 |
| | 6 | |

Then we can see that the array B can be obtained from B' which is

| 1 | a | b |
|---|---|---|
| 1 | 1 | |
| | | 1 |

by use of Lemma 2.2.1, that is by putting new integers into empty cells, which does not change S(B'). But S(B') = <a> = H.

2.4.3 Handling Coincidences.

2.4.3.1 Definition - A *coincidence* is defined to be the occurrence in an array of two integers which are equivalent, where we recall that i is equivalent to j if there is a path from i to j covering the identity.

The handling of coincidences is much the same as creating of deduction words. When a coincidence is found, for instance i is

equivalent to j, then when making the array consistent by replacing j by i where i < j, we concatenate the deduction word corresponding to (i is equivalent to j) to each new deduction word. This method will be illustrated in the following example. Instead of writing "i is equivalent to j" we shall write "i ~ j."

2.4.4 Example.

Consider $G = \langle a,b,c,d,e | abc^{-1} = bcd^{-1} = cde^{-1} = dea^{-1} = eab^{-1} = 1 \rangle$.

Let $H = \langle a \rangle$.

| 1 | a | b | c | d | e | |
|---|---|---|---|---|---|---|
| 1 | | | | | | |
| | 1 | | | | | Start with the identity. |
| | | 1 | | | | |
| | | | 1 | | | |
| | | | | 1 | | |
| | | | | | 1 | |

Creating deduction words as above leads to the array:

| 1 | a | b | c | d | e |
|---|---|---|---|---|---|
| 1 | 1 | 2 | $2_1$ | $3_2$ | $3_8$ |
| | | | 1 | | |
| | | | | 1 | |
| 2 | | | 3 | $3_9$ | $2_6$ |
| 3 | $2_7$ | $1_4$ | | $2_5$ | $1_3$ |
| | 3 | | | | |
| | | 3 | | | |

The deductions are,

#1 $1c2 = ab$,

#2 $1d3 = bc$,

#3 $3e1 = c^{-1}b^{-1}a$,

#4 $3b1 = c^{-1}b^{-1}a^2$,

#5 $3d2 = c^{-1}b^{-1}a^3b$,

#6 $2e2 = b^{-1}a^3b$,

#7 $3a2 = c^{-1}b^{-1}a^6b$,

#8 $1e3 = a^{-6}bc$, and

#9 $2d3 = b^{-1}a^{-7}bc$.

From 1d3 and 2d3 we get $1d3d^{-1}2 = (1d3)(3d^{-1}1)$

$= (bc)(b^{-1}c^{-1}a^7b) = a^7b$. Call this deduction #10, $(1 \sim 2) = a^7b$ .

But then (1b2)(2 ~ 1) gives us (b)($b^{-1}a^{-7}$) = $a^{-7}$ = (1b1). Call this deduction #11. But then (1c2)(2 ~ 1) gives us (ab)($b^{-1}a^{-7}$) = $a^{-6}$ = (1c1). Call this deduction #12. But then (1 ~ 2)(2c3) gives us ($a^{7}$b)(c) = $a^{7}$bc = (1c3). Call this deduction #13. Then we have (2$c^{-1}$1)(1c3) = ($b^{-1}a^{-1}$)($a^{7}$bc) = $b^{-1}a^{6}$bc = (2 ~ 3). Call this deduction #14. But then (1 ~ 2)(2 ~ 3) gives us ($a^{7}$b)($b^{-1}a^{6}$bc) = $a^{13}$bc = (1 ~ 3). Call this deduction #15. But then (1d3)(3 ~ 1) gives us (bc)($c^{-1}b^{-1}a^{-13}$) = $a^{-13}$ = (1d1). Call this deduction #16. But then (1e3)(3~1) gives us ($a^{-6}$bc)($c^{-1}b^{-1}a^{-13}$) = $a^{-19}$ = (1e1). Call this deduction #17.

Therefore the final table is:

| 1 | a | b | c | d | e |
|---|---|---|---|---|---|
| 1 | 1 | $1_{11}$ | $1_{12}$ | $1_{16}$ | $1_{17}$ |

Therefore G is cyclic generated by <a>. And we can see that each of the generators b,c,d,e is expressed in terms of of the generator a. Note also from (1~3)(3a2)(2~1) we have ($a^{13}$bc)($c^{-1}b^{-1}a^{6}$b)($b^{-1}a^{-7}$) = $a^{12}$ = (1a1). Call this deduction #18. Therefore a = $a^{12}$, so $a^{11}$ = 1, and G is a finite group.

This working is readily transcribed into a formal proof that the group is finite and cyclic. The numbers in the left margin indicate use of the numbered relation:

| | | |
|---|---|---|
| | $c = ab.$ | 1) |
| | $d = bc.$ | 2) |
| 2) | $e = d^{-1}a = c^{-1}b^{-1}a.$ | 3) |
| 3) | $b = ea = c^{-1}b^{-1}a^2.$ | 4) |
| 1),4) | $d=bc=c^{-1}b^{-1}a^3b.$ | 5) |
| 5) | $e=cd=cc^{-1}b^{-1}a^3b$ $= b^{-1}a^3b.$ | 6) |
| 5),6) | $a=de=c^{-1}b^{-1}a^6b.$ | 7) |
| 7) | $e=ba=bb^{-1}a^{-6}bc$ $=a^{-6}bc.$ | 8) |
| 1),8) | $d=c^{-1}e=b^{-1}a^{-1}a^{-6}bc$ $=b^{-1}a^{-7}bc.$ | 9) |
| 2),9) | $1=dd^{-1}=bcc^{-1}b^{-1}a^7b$ $=a^7b.$ | 10) |
| 10) | $b=b\cdot 1=bb^{-1}a^{-7}=a^{-7}.$ | 11) |

1),10) $c=c\cdot 1=ab\cdot 1=abb^{-1}a^{-7}$

$=a^{-6}$. 12)

10) $c=1\cdot c=a^{7}bc$. 13)

1),13) $1=c^{-1}c=b^{-1}a^{-1}a^{7}bc$ 14)

$=b^{-1}a^{6}bc$.

10),14) $1=1\cdot 1=a^{7}bb^{-1}a^{6}bc$ 15)

$= a^{13}bc$.

2),15) $d=d\cdot 1=bcc^{-1}b^{-1}a^{-13}$ 16)

$= a^{-13}$.

8),15) $e=e\cdot 1=a^{-6}bcc^{-1}b^{-1}a^{-13}$

$=a^{-19}$. 17)

7),15) $a = 1\cdot a\cdot 1=a^{13}bcc^{-1}b^{-1}a^{6}b^{-1}a^{-7}$

10) $=a^{12}$. 18)

Therefore $a = a^{12}$ so $a^{11} = 1$, the group is finite and cyclic, and we have expressed b,c,d,e in terms of <a>.

## 2.5 DIFFERENT IMPLEMENTATIONS OF COSET ENUMERATION

The Todd-Coxeter algorithm is a systematic procedure for enumerating cosets of a subgroup H of finite index in a group G, given

a set of defining relations for G and words generating H. At the present time, Todd-Coxeter programs represent the most common application of computers to group theory. They are used for constructing sets of defining relations for particular groups, for determining the order of a group from its defining relations, for studying the structure of particular groups, and for many other things.

We describe briefly four different implementations of the Todd-Coxeter coset enumeration algorithm.

### 2.5.1 Haselgrove - Leech-Trotter

The Haselgrove-Leech-Trotter method was developed by Haselgrove in 1953 and later adapted by Leech, Trotter, and others. In the Haselgrove-Leech-Trotter method, relators are applied to the cosets in the order in which the cosets were introduced. If for some coset i and relator r, the r-cycle at coset i is incomplete, sufficient new cosets are immediately introduced so as to complete the r-cycle at i.

### 2.5.2 Felsch Method

Suppose that the definition $i(s_j)=k$ has been made, where i, k

correspond to integers in the array, and $s_j$ is a generator for G. The Felsch procedure is to apply all significantly different cyclic permutations of relators beginning with $s_j$ to coset i. This process is repeated with any deductions $(i')(s_j') = k'$ which may have been discovered, until all possible consequences of the original definition have been discovered. Only at this stage will a new coset be introduced, if necessary, and then by defining it so that the first vacant position in the coset table is filled.

2.5.3 Guy - Lookahead

A type of lookahead program was used by Leech in 1959, but this form of the Todd-Coxeter algorithm did not really begin to develop until Guy wrote a lookahead Todd-Coxeter program on the ATLAS at Cambridge in 1967.

The lookahead method operates in two distinct phases: a defining phase and a lookahead phase. As long as the number of cosets defined at any instant is less than a certain specified number $M_L$, the algorithm remains in the defining phase. In this phase, the

enumeration proceeds by the Haselgrove-Leech-Trotter method. When, however, the number of cosets defined exceeds the limit $M_L$, the algorithm switches to the lookahead phase. Here, relators are applied to cosets as before, but if a relator cycle is incomplete at some coset, no new cosets are defined to complete the cycle. The aim is to discover a large number of deductions and coincidences without introducing any new cosets. If enumeration is still incomplete at the end of the lookahead phase and if sufficient storage space is available, we return to the definition phase in which we remain until either the enumeration is complete or the number of cosets defined again passes some preset limit. Thus, the algorithm alternates between the defining phase and the lookahead phase.

2.5.4 Cannon, Dimino, Havas, Watson - Modified Lookahead

There are a number of ways of arranging the lookahead. Guy, for example, divides his available space into a number of blocks and applies lookahead before allowing the coset table to extend into a new block. We shall call this technique bumping.

If the optimum block size is chosen, this technique will result in

extremely rapid enumerations. On the other hand, a poor choice of block size can result in inefficient enumerations. The other possibility, is to let the coset table exhaust the available space before applying lookahead. Also, when the program is in the lookahead phase, it can return to the defining phase as soon as a single coincidence has been discovered (incremental lookahead) or only after all relators have been applied to all cosets currently defined (complete lookahead).

If all cosets have been processed in the lookahead phase of an incremental lookahead program, the program begins again with the first coset not yet processed in the defining phase. A single application of lookahead, of either kind, to every coset not yet processed in the defining phase is called a lookahead pass. Generally, both incremental and complete lookahead programs are arranged so that they terminate when either the coset enumeration completes or a lookahead pass fails to discover a single coincident coset. In the case of complete lookahead, a considerable amount of time can be saved in situations where the enumeration does not complete by specifying

that execution is to terminate if less than a certain number of coincidences are discovered during an application of lookahead.

The lookahead program can be modified slightly to allow the lookahead phase to use relators (redundant relators) which are not used in the defining phase.

## 2.6 LIMITATIONS OF COSET ENUMERATION

The obvious limitations of coset enumeration are time and space. Only a finite amount of computer space is available for storage; one cannot always enumerate every coset of a particular subgroup even if the subgroup has finite index.

The central problem in programming the Todd-Coxeter algorithm is finding a satisfactory rule for introducing new cosets. As the range of application of a Todd-Coxeter program is thus limited by the amount of storage required to hold the partial coset tables generated during an enumeration, one tries to define cosets in such a way that as few redundant cosets as possible are introduced.

It is easily seen, however, that the application of the Todd-Coxeter algorithm to certain presentations will necessarily

require the introduction of redundant cosets.

For example the group $G=\langle a \mid a^p = a^q = 1, p, q \text{ prime } p \neq q \rangle$ is trivial but min(p,q) cosets must be defined before the relation a=1 is discovered. Given any integer m, one can produce a presentation for the trivial group which requires the definition of at least m cosets before the Todd-Coxeter algorithm is able to deduce the group is trivial. Then by adding such a presentation of the trivial group to some presentation of a group G, we can produce an arbitrarily bad presentation for G.

## 2.7 ORIGINAL TODD-COXETER COSET ENUMERATION

Let $G = \langle X \mid R \rangle$ be a finite group and put $F = \langle X \mid \rangle$, $N = [R]$, so that G is isomorphic to F/N; then to perform a coset enumeration on G is simply to count cosets of N in F, that is to find |G|. The first coset enumeration method was developed in 1936 and many refinements have been introduced since that time, particularly since the advent of high-speed computing machines, to which the method is readily adaptable. The version given here is the original one, due to Todd and Coxeter.

For each relator $r = x_1 \ldots x_n \in R$, with $x_1 \ldots x_n$ a reduced word in $X \cup X^{-1}$, we draw a rectangular table having n+1 rows and a certain number (for the moment unlimited) of rows. We begin by entering the symbol '1' in the first and last places of the first row of each table, the remaining places in the first row being as yet empty. We then fill some empty space with the symbol '2' (usually next to some '1' -- either to the left or right of it). Suppose the situation to be that a 2 is to the right of 1 with $x_1 \in X \cup X^{-1}$ lying between them in the first row of the table. Then we put a 2 in the first and last places of the second row of each table and, wherever in any table 1 lies to the left of an empty space with $x_1$ between the two spaces or to the right of an empty space with $x_1^{-1}$ between the spaces, we fill that empty space with a 2. Similarly, if 2 lies to the right (left) of an empty space with $x_1$ ($x_1^{-1}$) between the spaces, we fill that space with a 1. The idea behind the process (which we call scanning) is that 1 and 2 correspond to the elements 1 and $x_1$ of G, so that we may write

$1x_1 = 2$, and $2x_1^{-1} = 1$.

Having made sure that no more spaces can be filled in this way, we fill an empty space with the new symbol 3, begin a new row of the tables and scan as above, entering all possible 1's, 2's and 3's in accordance with definitions of 2 and 3. Then fill an empty space with the symbol 4, begin the fourth row of the tables and scan again. The process terminates when there are no more empty spaces, whereupon the total number of rows is equal to |G|. There are two ways in which we obtain information of the type ix = j other than by definition. The first is when any row of a table becomes complete. Thus in the transition $| \, | \, x_i \, | \quad | \, x_{i+1} | \, j \;\rightarrow\; | \, | \, x_i \, | \, k \, | \, x_{i+1} \, | \, j$ we obtain two pieces of information, namely $lx_i = k$, $kx_{i+1} = j$; one of these was either known already or was a definition, the other may be regarded as a bonus. The second way is when we obtain information, say ix = j, when we already know ix = k. We then deduce j = k and that the corresponding entries in the j and k rows of all the tables are equal. This situation is called *coset collapse*, and when it occurs we delete the kth row if k > j. Each row of each table begins and ends with the

same symbol because the word at the head of each table is a relation holding in G.

Incorporated in the original exposition of Todd and Coxeter is a valuable refinement of the process, whereby we enumerate the cosets of a nontrivial subgroup, H say, of G; the resulting number of rows is the index | G : H |. The subgroup H is specified by giving its generators as reduced words in $X \cup X^{-1}$ and the process is the same as above, with the addition of one-rowed tables (beginning and ending with 1) for each generator of H. The process terminates when all tables are complete.

It may be clear already why the orginal coset enumeration algorithm of Todd and Coxeter is equivalent to the algorithm given in Sections 2.1 and 2.2.

Completing the one-rowed tables corresponding to generators of H corresponds to creating paths from 1 to 1 covering generators of H, which is the first step of the algorithm given in Sections 2.1 and 2.2. Completing the tables with relators at the head of each table is equivalent to processing integers in the algorithm given in Sections

2.1 and 2.2. When we process an integer, say $k > 0$, then for each relator $r_j$, we create a path from k to k covering $r_j$. This corresponds to completing a row of each relator table that has the integer k at the beginning and end of the row. As the original Todd and Coxeter algorithm terminates when all tables are complete, that is have no empty spaces, the algorithm given in Sections 2.1 and 2.2 terminates when all integers have been processed.

## 2.8 COMPUTER IMPLEMENTATION

As mentioned previously, the coset enumeration algorithm is mechanical enough to be implemented on a computer. The following is a discussion of such an implementation. The computer program corresponds to the algorithm as described in Sections 2.1 and 2.2.

The computer program needs the following subroutines and functions:

1. READPRESENTATION() - a subroutine to read a presentation of G;
2. READSUBGROUP() - a subroutine that reads the subgroup H and its generators;
3. FILLCELL(I,J) - a subroutine to fill in the empty cell (I,J) with an

integer not occuring in the array A according to Lemma 2.2.1;

4. REPLACEINTEGER(P,Q) - a subroutine to replace an integer Q>P by P throughout an array A according to Lemma 2.2.2;

5. MAKECONSISTENT() - a subroutine to make a finite array A consistent according to Lemma 2.2.4;

6. CREATEPATH(K,W) - a subroutine which creates a path from an integer K > 0 to K covering word W. This subroutine may make use of subroutines FILLCELL(I,J), REPLACEINTEGER(P,Q), and MAKECONSISTENT();

7. CREATEIDENTITY() - a subroutine to create an initial array A with n+1 rows and n+1 columns with 1's along the main diagonal and empty cells everywhere else. This amounts to creating an array A such that S(A) = <1>. We are assuming that G = <X | R>, X = $\{x_1, \ldots, x_n\}$;

8. PROCESSINTEGER(K) - a subroutine which for each relator $r_j$, creates a path from K to K covering $r_j$, where K > 0. This subroutine would make use of CREATEPATH(K,$R_J$). This subroutine also takes an array A and fills all empty cells of the row with K in column 0, and all

empty cells in column 0 which have a K in that row by use of subroutine FILLCELL(I',J');

9. NEXTUNPROCESSEDINTEGER() - a function which determines the next unprocessed integer of an array A. The function returns an integer K>0 as the next unprocessed integer or 0 if all integers in the array A have been processed;

10. NUMBEROFGENERATORS(H) - a function which returns an integer corresponding to the number of generators of the subgroup H;

11. OUTOFMEMORY() - a function which determines if we are running out of memory. This function returns TRUE if we are almost out of memory, and FALSE otherwise;

12. PRINTARRAY - a subroutine to print the current array A;

13. NUMBEROFROWS() - A function which returns the number of rows of the current array A. Hence if A is consistent and complete this function returns the index of H in G.

The main body of the program would look like the following:

```
PROGRAM COSETENUMERATION();
{
      READPRESENTATION();
      READSUBGROUP();
      CREATEIDENTITY();
      FOR I = 1 TO NUMBEROFGENERATORS(H)
      {
            CREATEPATH(1,H(I));

/*          H(I) IS A GENERATOR OF H.  CREATE A PATH FROM 1
      TO 1 COVERING H(I) FOR EACH GENERATOR H(I) OF H */
      }
      LET K = 1;
      LET DONE = FALSE;
      WHILE NOT DONE DO
      {
            PROCESSINTEGER(K);
/*          CREATE PATH FROM K TO K COVERING EACH
                                    RELATOR R_J  */
            LET K = NEXTUNPROCESSEDINTEGER();
            IF K = 0 THEN LET DONE = TRUE;
            IF OUTOFMEMORY() THEN LET DONE = TRUE;
      }
      IF K = 0 THEN
      {
            PRINTARRAY();
            PRINT("THE INDEX OF H IN G IS ", NUMBEROFROWS());
      }
      ELSE
      {
            PRINT("UNABLE TO COMPLETE ARRAY.  RUNNING OUT OF
            MEMORY.");
            PRINTARRAY();
      }
}
```

The details of programming each subroutine and function are left to the reader.

# CHAPTER III

# PRESENTATIONS OF SUBGROUPS

Although implicit in the work of Reidemeister, the process described below has only recently been exploited. Like that of coset enumeration, its popularity is in the development of high-speed computers. There are many programs adapting both methods for machine computation, and at least one which combines the two in a rather elegant way.

## 3.1 STATEMENT OF THE PROBLEM

Our object is to produce a presentation for a given subgroup H of finite index in a group G for which a finite presentation

$G = \langle X \mid R \rangle$,

$X = \{x_1, \ldots, x_n\}$ $R = \{r_1, \ldots, r_m\}$ (1)

is known.

The subgroup H is given as the subgroup of G generated by some finite subset Y of G, where $Y = Y(X)$, $Y = \{y_1, \ldots, y_l\}$ (2), that is, the $y_i$ are expressed as words in $X \cup X^{-1}$.

3.2 THE METHOD

The method in principle is very simple and proceeds in five steps.

Step 1. Find the transversal U. Let C be the preimage of H under the natural map

$v: F \rightarrow G$

$X \rightarrow X$

where $F = \langle X \mid \rangle$. By enumerating cosets, we find a right transversal $U = \{u_1, \ldots, u_g\}$ for C in F. Since $H = C/[R]$, we can regard U as a transversal for H in G. We assume U has the Schreier property (Lemma 1.2.2).

Step 2. In accordance with Theorem 1.2.5, the set

$B = \{ux(\tau(ux))^{-1} : u \in U, x \in X\} \setminus \{1\}$ forms a set of free generators for C. Write $B = \{b_1, \ldots, b_k\}$ where $k = (n-1)g+1$.

Step 3. Since $U^{-1}$ is a left transversal for C in F, any element $f \in F$

can be written in the form $f = u^{-1}c$, $u \in U$, $c \in C$. Now [R], being the normal closure of R in F, is generated by elements of the form $f^{-1}\, r\, f$, $r \in R$, $f \in F$, and $f^{-1}\, r\, f = c^{-1}\, u\, r\, u^{-1}\, c = c^{-1}(u\, r\, u^{-1})\, c$ and so [R] is the normal closure in C of the set $S = \{u_j\, r_i\, u_j^{-1} : 1 \leq i \leq m,\ 1 \leq j \leq g\}$ of cardinality mg.

Step 4. We now have generators B and relators S = S(X) for H.

Using Lemma 1.2.3, each element of S can be expressed in terms of the $b_j$, say S = S(B), whereupon $H = \langle B \mid S(B) = 1 \rangle$ is a finite free presentation for H.

Step 5. If the generators for H are prescribed as in (2), we use Lemma 1.2.3 to obtain Y = Y(B). Then using enhanced coset enumeration (Leech) of Section 2.4, we obtain B = B(Y), that is, we express the elements of B in terms of the generators Y of H.

We then use Tietze transformations in Theorem 1.4.4 to obtain $H = \langle Y \mid S(B(Y)) = 1,\ Y = Y(B(Y)) \rangle$.

3.3 EXAMPLES

3.3.1 Let $G = \langle x,y \mid x^3 = y^2 = (xy)^2 = 1 \rangle$, $H = \langle y \rangle$.

Find the presentation for H in terms of the generators for H.

Step 1. First perform enhanced coset enumeration to obtain

| 1 | x | y |
|---|---|---|
| 1 | 2 | 1 |
| 2 | 3 | $2_2$ |
| 3 | $1_1$ | $2_3$ |

where deductions are

#1 $3x1 = 3x^{-1}2x^{-1}1$,

#2 $2y3 = 2x^{-1}1y^{-1}1x2x3$, and

#3 $3y2 = 3x^{-1}2x^{-1}1y1x2$.

Since $1 \cdot 1 = 1$, $1x = 2$, $1y=1$, and $1x^{-1} = 3$, we obtain the Schreier

transversal $U = \{1,x,x^{-1}\}$.

Step 2.

We have $B = \{ux ( \tau (ux))^{-1} : u \in U, x \in X\}$ where $U = \{1,x, x^{-1}\}$, $X = \{x,y\}$.

Therefore we have

$1x ( \tau (1x))^{-1} = x x^{-1} = 1$,

$1y( \tau (1y))^{-1} = y$,

$xx( \tau (xx))^{-1} = x^3$,

$xy(\tau\ (xy))^{-1}=xyx$,

$x^{-1}x(\tau\ (x^{-1}x))^{-1} = 1$, and

$x^{-1}y(\tau\ (x^{-1}y))^{-1} = x^{-1}yx^{-1}$.

Therefore $B = \{y, x^3, xyx, x^{-1}yx^{-1}\}$ is a free set of generators for C of order (2-1)3 + 1 = 4. Let $B = \{b_1,b_2,b_3,b_4\}$.

Step 3.

We have $S = \{u_j\ r_i\ u_j^{-1} : 1 \le i \le m,\ 1 \le j \le m\}$ where $U = \{1,x,x^{-1}\}$ and $R = \{ x^3, y^2, (xy)^2\}$. Therefore $S = \{x^3, y^2, xy^2x^{-1}, x^{-1}y^2x, (xy)^2, x(xy)^2x^{-1}, yxyx\}$.

Step 4. Express the elements of S in terms of B. We have

$x^3 = x^3 = b_1$,

$y^2 = (y)(y) = b_1^2$,

$xy^2x^{-1} = (xyx)(x^{-1}yx^{-1}) = b_3b_4$,

$x^{-1}y^2x = (x^{-1}yx^{-1})(xyx) = b_4b_3$,

$xyxy = (xyx)(y) = b_3b_1$,

$x\ (xyxy)\ x^{-1} = (x^3)\ (x^{-1}yx^{-1})(x^3)\ (x^{-1}yx^{-1}) = b_2b_4b_2b_4$, and

$yxyx = (y)(xyx) = b_1b_3$.

Therefore $H = \langle b_1, b_2, b_3, b_4 \mid b_2$

$$= b_1^2 = b_3b_4 = b_4b_3 = b_3b_1 = b_2b_4b_2b_4 = b_1b_3 = 1\rangle$$

where $b_1 = y$, $b_2 = x^3$, $b_3 = xyx$, $b_4 = x^{-1}yx^{-1}$.

Step 5. Express B in terms of the generators of H using enhanced coset enumeration. We have

$y = b_1$,

$b_1 = y = y$,

$b_2 = x^3 = 1x2x3x1 = x^2x^{-2} = 1$,

$b_3 = xyx = 1x2y3x1 = xx^{-1}y^{-1}xxx^{-1}x^{-1} = y^{-1}$, and

$b_4 = x^{-1}yx^{-1} = 1x^{-1}3y2x^{-1}1 = xxx^{-1}x^{-1}yxx^{-1} = y$.

Therefore $H = \langle y \mid 1 = y^2 = y^{-1}y = yy^{-1} = y^2 = yy^{-1} = 1, y = y\rangle$ or simplifying we have $H = \langle y \mid y^2 = 1\rangle$ as expected.

3.3.2 Let $G = \langle x, y, x^2 = y^4 = xyx^{-1}y = 1\rangle$, $H = \langle xy\rangle$.

Find a presentation for H in terms of the generators of H.

Step1. Find a Schreier transversal.

Using enhanced coset enumeration, we have

| 1 | x | y |
|---|---|---|
| 1 | 2 | 3 |
| 2 | $1_1$ | 1 |
| 3 | $4_4$ | 4 |
| 4 | $3_3$ | $2_2$ |

where deductions are

#1 $2x1 = 2x^{-1}1$,

#2 $4y2 = 4y^{-1}3y^{-1}1y^{-1}2$,

#3 $4x3 = 4y^{-1}3y^{-1}1y^{-1}2x^{-1}1y3$, and

#4 $3x4 = 3y^{-1}1x2y1y3y4$.

Because $1 \cdot 1 = 1$, $1x=2$, $1y=3$, $1x^{-1}= 2$, $1y^{-1}=2$, $1xx=1$, $1xy=1$, and $1xy^{-1}=4$ we have a Schreier transversal $U = \{1,x,y,xy^{-1}\}$.

Step 2. Find the set of free generators B for C.

We have $B = \{ ux ( \tau (ux))^{-1} : u \in U, x \in X\}$ where

$U = \{1,x,y, xy^{-1}\}$, $X = \{x,y\}$. Therefore we have

$1x ( \tau (1x))^{-1} = xx^{-1}=1$,

$1y ( \tau (1y))^{-1} = yy^{-1} = 1$,

$xx\,(\tau\,(xx))^{-1} = x^2 = x^2$,

$xy\,(\tau\,(xy))^{-1} = xy$,

$yx(\tau\,(yx))^{-1} = yxyx^{-1}$,

$yy(\tau\,(yy))^{-1} = y^3x^{-1}$,

$xy^{-1}x\,(\tau\,(xy^{-1}x))^{-1} = xy^{-1}xy^{-1}$, and

$xy^{-1}y\,(\tau\,(xy^{-1}y))^{-1} = 1$.

Therefore $B = \{x^2, xy, yxyx^{-1}, y^3x^{-1}, xy^{-1}xy^{-1}\}$ forms a free set of generators for C of order (2-1)4 + 1 = 5. Let $B = \{b_1,b_2,b_3,b_4,b_5\}$.

Step 3. Find a set of relators for H.

We have $S = \{u_j\, r_i\, u_j^{-1}: 1 \le i \le m, 1 \le j \le g\}$ where $U = \{1,x,y, xy^{-1}\}$, and $R = \{x^2, y^4, xyx^{-1}y\}$.

So $S = \{x^2,y^4,xyx^{-1}y,xy^4x^{-1},xxyx^{-1}yx^{-1},yx^2y^{-1},yxyx^{-1},$

$xy^{-1}x^2yx^{-1},xy^{-1}xyx^{-1}yyx^{-1}\}$.

Step 4. Express S in terms of B. We have

$x^2 = x^2 = b_1$,

$y^4 = (y^3x^{-1})(xy) = b_4b_2$,

$xyx^{-1}y = (xy)(x^2)^{-1}(xy) = b_2b_1^{-1}b_2$,

$xy^4x^{-1} = (xy)(y^3x^{-1}) = b_2b_4$,

$x^2yx^{-1}yx^{-1} = (x^2)(xy^{-1}xy^{-1})^{-1} = b_1b_5^{-1}$,

$yx^2y^{-1} = (yxyx^{-1})(xy^{-1}xy^{-1}) = b_3b_5$,

$yxyx^{-1} = yxyx^{-1} = b_3$,

$xy^{-1}x^2yx^{-1} = (xy^{-1}xy^{-1})(yxyx^{-1}) = b_5b_3$, and

$xy^{-1}xyx^{-1}y^2x^{-1} = (xy^{-1}xy^{-1})(y^3x^{-1})(yxyx^{-1})^{-1}(y^3x^{-1}) = b_5b_4b_3^{-1}b_4$.

So $H = \langle b_1, b_2, b_3, b_4, b_5 \mid b_1 = b_4b_2 = b_2b_1^{-1}b_2 = b_2b_4 = b_1b_5^{-1}$

$= b_3b_5 = b_3 = b_5b_3 = b_5b_4b_3^{-1}b_4 = 1 \rangle$.

Step 5. Using enhanced coset enumeration, express B in terms of Y.

We have

$b_1 = x^2 = 1x2x1 = xx^{-1} = 1$,

$b_2 = xy = 1x2y1 = xy$,

$b_3 = yxyx^{-1} = 1y3x4y2x^{-1}1 = y(y^{-1}xy^3)(y^{-3})\,x^{-1} = 1$,

$b_4 = y^3x^{-1} = 1y3y4y2x^{-1}1 = y\,y\,(y^{-3})x^{-1} = (xy)^{-1}$, and

$b_5 = xy^{-1}xy^{-1} = 1x2y^{-1}4x3y^{-1}1 = x(y^3)(y^{-3}x^{-1}y)y^{-1} = 1.$

So $H = \langle(xy)| 1 = (xy)^{-1}(xy) = (xy)(xy) = (xy)(xy)^{-1} = 1=1=1=1$

$=(xy)^{-2} = 1, xy = xy\rangle,$

or $H = \langle(xy) \mid (xy)^2 = 1\rangle$.

3.3.3 Let $G = \langle x,y \mid x^3 = y^3 = 1\rangle$, $H = \langle xy,yx\rangle$.

Find a presentation for H in terms of the generators of H.

Step 1. Find the Schreier transversal for H.

Using enhanced coset enumeration we have

| 1 | x | y |
|---|---|---|
| 1 | 2 | 3 |
| 2 | $3_1$ | 1 |
| 3 | 1 | $2_2$ |

where deductions are

#1 $2x3 = 2x^{-1}1x^{-1}3$, and

#2 $3y2 = 3y^{-1}1y^{-1}2$.

Since $1\cdot1 = 1$, $1x = 2$, and $1y = 3$, the Schreier transversal U is

$U = \{1,x,y\}$.

Step 2. Find a free set of generators B for C.

We have $B = \{ux(\tau(ux))^{-1} : u \in U, x \in X\}$ where $U = \{1,x,y\}$, $X = \{x,y\}$.

Therefore we have

$1x(\tau(1x))^{-1} = 1$,

$1y(\tau(1y))^{-1} = 1$,

$x^2(\tau(x^2))^{-1} = x^2y^{-1}$,

$xy(\tau(xy))^{-1} = xy$,

$yx(\tau(yx))^{-1} = yx$, and

$y^2(\tau(y^2))^{-1} = y^2x^{-1}$.

Therefore $B = \{x^2y^{-1}, xy, yx, y^2x^{-1}\}$ is a set of free generators for C of order $3(2-1)+1 = 4$. Let $B = \{b_1,b_2,b_3,b_4\}$.

Step 3. Find a set of relators for H.

We have $S = \{u_jr_iu_j^{-1} : 1 \le i \le m, 1 \le j \le g\}$. Therefore $S = \{ x^3, y^3, xy^3x^{-1}, yx^3y^{-1}\}$ is a set of relators for H.

Step 4. Express S in terms of B. We have

$x^3 = (x^2y^{-1})(yx) = b_1b_3$,

$y^3 = (y^2x^{-1})(xy) = b_4b_2$,

$xy^3x^{-1} = (xy)(y^2x^{-1}) = b_2b_4$, and

$yx^3y^{-1} = (yx)(x^2y^{-1}) = b_3b_1$.

Therefore $H = \langle b_1,b_2,b_3,b_4 | \ b_1b_3 = b_4b_2 = b_2b_4 = b_3b_1 = 1\rangle$.

Step 5. Express B in terms of Y. We have

$b_1 = x^2y^{-1} = 1x2x3y^{-1}1 = x(x^{-2})\ y^{-1} = (yx)^{-1}$,

$b_2 = xy = 1x2y1 = xy$,

$b_3 = yx = 1y3x1 = yx$, and

$b_4 = y^2x^{-1} = 1y3y2x^{-1}1 = y(y^{-2})x^{-1} = (xy)^{-1}$,

Therefore $H = \langle (xy),(yx) | (yx)^{-1}(yx) = (xy)^{-1}(xy) = (xy)(xy)^{-1}$

$= (yx)(yx)^{-1} = 1,\ xy=xy,\ yx=yx\rangle$,

or $H = \langle (xy),\ (yx) | \ \rangle$ and so H is a free group.

3.3.4 Let $G = \langle x_1, \ldots, x_n | \ x_i^2 = 1,\ (x_ix_j)^{m_{ij}} = 1,\ i = 1, \ldots, n,\ j = 1, \ldots, n\rangle$,

where $m_{ij}$ is a positive integer for all $i,j = 1, \ldots, n$.

Let $H = \text{Ker}\ \Phi$ where $\Phi$ is a homomorphism from G to $\langle \{0,1\} + \rangle$

such that $\Phi(x_i) = 1$ for $i = 1, \ldots, n$. Find a presentation for H.

Step 1. Find a Schreier transversal U for C. Coset enumeration

produces

| $\underline{1}$ | $\underline{x_1}$ | $\cdots$ | $\underline{x_n}$ |
|---|---|---|---|
| 1 | 2 | | 2 |
| 2 | 1 | | 1 |

Since $1 \cdot 1 = 1$, and $1x_1 = 2$, a Schreier transversal U is $U = \{1, x_1\}$.

Step 2. Find the set B of free generators for C.

We have $B = \{ux(\tau\ (ux))^{-1} : u \in U, x \in X\}$ where

$U = \{1, x_1\}$, $X = \{x_1, \ldots, x_n\}$. Therefore we have

$1x_1\ (\tau\ (1x_1))^{-1} = x_1x_1^{-1} = 1$, $\qquad x_1x_1(\tau\ (x_1x_1))^{-1} = x_1^2 = b_n$,

$1x_2\ (\tau\ (1x_2))^{-1} = x_2x_1^{-1} = b_1$, $\qquad x_1x_2(\tau\ (x_1x_2))^{-1} = x_1x_2 = b_{n+1}$,

. .

. .

. .

$1x_n(\tau\ (1x_n))^{-1} = x_nx_1^{-1} = b_{n-1}$, and $x_1x_2(\tau\ (x_1x_n))^{-1} = x_1x_n = b_{2n-1}$.

Therefore $B = \{b_1, b_2, \ldots, b_{2n-1}\}$ forms a free set of generators for C of order $(n-1)2 + 1 = 2n-1$.

Step 3. Find a set of relators S for H.

We have $S = \{u_j r_i u_j^{-1} : 1 \le i \le m, 1 \le j \le g\}$ where $U = \{1, x_1\}$ and $R = \{ x_i^2, (x_i x_j)^{m_{ij}} : i = 1, \ldots, n, j = 1, \ldots, n\}$. So

$S = \{ x_i^2, (x_i x_j)^{m_{ij}}, x_1(x_i^2)x_1^{-1}, x_1(x_i x_j)^{m_{ij}}x_1^{-1} : i = 1, \ldots, n, j=1, \ldots, n\}$.

Step 4. Express S in terms of B. We have

$x_i^2 = b_{i-1}b_{n+i-1}$, $i = 1, \ldots, n$, $b_0 = 1$,

$(x_i x_j)^{m_{ij}} = (b_{i-1}b_{n+j-1})^{m_{ij}}$, $i = 1, \ldots, n$, $j = 1, \ldots, n$, $b_0 = 1$,

$x_1(x_i^2)x_1^{-1} = b_{n+i-1}\, b_{i-1}$, $i =1, \ldots, n$, $b_0 = 1$, and

$x_1(x_i x_j)^{m_{ij}}x_1^{-1} = (b_{n+i-1}b_{j-1})^{m_{ij}}$, $i = 1, \ldots, n$, $j = 1, \ldots, n$, $b_0 = 1$.

Therefore $H = \langle b_1, \ldots, b_{2n-1} | \; b_{i-1}b_{n+i-1} = b_{n+i-1}b_{i-1} = (b_{i-1}b_{n+j-1})^{m_{ij}}$

$= (b_{n+i-1}b_{j-1})^{m_{ij}} = 1$, $i = 1, \ldots, n$, $j = 1, \ldots, n$, $b_0 = 1\rangle$.

Since $b_{n+i} = b_i^{-1}$ for $i = 0, \ldots, n-1$, where $b_0 = 1$, we have

$b_{n+i-1} = b_{i-1}^{-1}$ for $i = 1, \ldots, n$.

Therefore $H = \langle b_1, \ldots, b_{n-1} \mid b_{i-1}b_{i-1}^{-1} = b_{i-1}^{-1}b_{i-1} = (b_{i-1}b_{j-1}^{-1})^{m_{ij}}$

$= (b_{i-1}^{-1}b_{j-1})^{m_{ij}} = 1,\ i = 1, \ldots, n,\ j = 1, \ldots, n,\ b_0 = 1\rangle,$

which simplifies to

$H = \langle b_1, \ldots, b_{n-1} \mid (b_{i-1}b_{j-1}^{-1})^{m_{ij}} = 1,\ i = 1, \ldots, n,\ j = 1, \ldots, n,\ b_0 = 1\rangle.$

Note the $b_i$'s are defined above as words in $X \cup X^{-1}$.

## BIBLIOGRAPHY


[1] J. Cannon, L. Dimino, G. Havas, J. Watson, "Implementation and Analysis of the Todd-Coxeter Algorithm," Mathematics of Computation, v. 27, 1973, pp. 463-490.

[2] H. S. M. Coxeter and J. A. Todd. "A Practical Method for Enumerating Cosets in an Abstract Finite Group," Proc. Edinburgh Math. Soc. (2), 5, 1936.

[3] L. A. Dimino, "A Graphical Approach to Coset Enumeration," SIGSAM Bulletin, v. 19, 1971, pp. 8 - 43.

[4] D. L. Johnson. Presentations of Groups, Cambridge University Press, Cambridge, 1976.

[5] J. Leech, "Computer Proof of Relations in Groups," in Topics in Group Theory and Computation (Edited by M. P. J. Curran), Academic Press, London and New York, 1977, pp. 38-61.

[6] J. Leech, "Coset Enumeration" in Computational Problems in Abstract Algebra (Edited by John Leech), Pergamon Press, New York and Oxford, 1970, pp. 21 - 35.

[7] I. D. Macdonald. The Theory of Groups, Oxford University Press, Oxford, 1968.

[8] W. Magnus, A. Karrass and D. Solitar. Combinatorial Group Theory, Interscience, New York, 1966.

[9] N. S. Mendelsohn, "An Algorithmic Solution for a Word Problem in Group Theory," Canad. J. of Math., v. 16, 1964, pp. 509-516.

[10] Michio Suzuki. Group Theory I, Springer-Verlag, New York, 1982.

[11] H. F. Trotter, "A Machine Program for Coset Enumeration," Canad. Math. Bull., v. 7, 1964, pp. 357-368.