Graduate Seminar - 7 May 2014

Daniel Rogers

May 5, 2014

1 Introduction

All groups we will consider in this talk will be finite. We multiply permutations from left to right, so $(1,2)(1,3) = (1,2,3)$, not $(1,3,2)$.

Definition 1.1. A *presentation* of a group is a pair $\langle X|R\rangle$ where X is a list of *generators* of the group, and R a list of *relations* which they satisfy.

Informally a presentation defines the largest group which satisfies those conditions. They are generally not difficult to quickly get an intuitive understanding of.

Example 1.2.
- Let $n \in \mathbb{N}$; then $\langle x|x^n = 1\rangle$ is a presentation of the cyclic group C_n.
- Let D_{2n} denote the dihedral group of order $2n$; in other words the rotations of a regular n-gon. (Draw a picture). This has a presentation given by $\langle r,s|r^n = s^2 = (rs)^2 = 1\rangle$.

We can find a presentation for any finite group G by letting $X = G$ and R be the set of all true relations, although this isn't generally useful.

Presentations of groups are often the most compact way of describing a group and can be informative of the structure of the group. Groups given by presentations often occur naturally in some situations; for instance it is often the easiest way to describe all groups of a given small order (e.g if it is a semidirect product), and these also occur in the wild, for example in some aspects of algebraic topology.

2 Base Strong Generating Sets

Question: suppose we have a group G and a subgroup $H \leq G$. How do we decide, given $g \in G$, whether $g \in H$? We can answer this question using the action of the groups G and H on a suitable set

Throughout this section, let G be a group acting on a set Ω. In this talk we will generally consider $\Omega = \{1,...,n\}$ and $G \leq \mathrm{Sym}(\Omega)$.

Definition 2.1. A *base* for G is a sequence $B = [b_1,...,b_m] \subset \Omega$ such that for $g \in G$, $b_i^g = b_i \ \forall \ i \in [1,...,n] \iff g = 1_G$. Elements of B are called *base points*, and are indexed by their position in the sequence; in particular we may talk of the i^{th} *base point* of G.

Example 2.2. Let $G = \mathrm{Sym}(4)$ acting on the set $\Omega = \{1,2,3,4\}$. The smallest base we can obtain here is any 3-point subset of Ω. For instance, take $B = [4,3,2]$. Then any $g \in G$ such that $4^g = 4, 3^g = 3, 2^g = 2$ must also satisfy $1^g = 1$ and so $g = \mathrm{id}$.

Definition 2.3. The *basic stabilizer chain* associated with a base B for G is the chain given by

$$G \geq G_{(b_1)} \geq G_{(b_1,b_2)} \geq ... \geq G_{(b_1,...,b_n)} = 1$$

We use the notation $G^{(i)} := G_{\{b_1,...,b_i\}}$ and $G^{(0)} = G$ to simplify notation for the basic stabilizer chain.

Example 2.4. In the above case, we have the chain $G^{(0)} = \langle(1,2,3,4),(1,2)\rangle \geq G^{(1)} = \langle(1,2,3),(1,2)\rangle \geq G^{(2)} = \langle(1,2)\rangle \geq G^{(3)} = \langle \mathrm{id}\rangle$.

Definition 2.5. A *strong generating set* for G relative to a base B is a sequence $S \subseteq G$ such that $\forall i \in [0,...,n]$, $\langle S \cap G^{(i)}\rangle = G^{(i)}$.

Thus, a strong generating set is a generating set for G that also contains generators for each group in the basic stabilizer chain.

Example 2.6. In the above case, from the way we have constructed the basic stabiliser chain it is clear that $S = \{(1,2,3,4),(1,2,3),(1,2)\}$ is a strong generating set for G.

Note that there is no requirement for a base or a strong generating set to be minimal, although in practice we seek to avoid superfluous base points or strong generators.

Example 2.7. (Optional)

- Let $G \leq S_8$ be given by $G = \langle(1,2)(5,6)(7,8),(3,4)(5,7)(6,8)\rangle \cong C_2 \times C_2$. Both $\{1,3\}$ and $\{5\}$ are bases for G, and both are minimal in the sense that no base point is redundant. Thus the size of a given base, even if the base is minimal, does not give a well-defined notion of the dimension of a group. The minimal size of a base of G is an object which can be of interest to study.

- Given any matrix group G viewed as acting on the set $\Omega = \mathbb{F}^n$, it is clear that one possible choice for a base of G is a basis of Ω. However, others (e.g. Butler) have considered the action of G on one-dimensional subspaces of Ω, which served to reduce the size of the basic orbits. The disadvantage of this method is that the action of G on the collection of subspaces of Ω may not be faithful, so we may not be able to find a base consisting entirely of subspaces. To avoid this problem, we often follow each subspace of Ω with a nontrivial vector contained in the subspace. This increases the size of the base but decreases the size of the orbits.

Definition 2.8. A *base strong generating set (BSGS)* for a group G is, as the name suggests, a pair (B,S) of a base of the group G and an associated strong generating set.

Definition 2.9. (Optional) A *strong presentation* for a group G with a BSGS (B,S) is a presentation $\langle S|R\rangle$ on the strong generating set of G, with the additional property that within this presentation we have subpresentations for each of the entries in the basic stabilizer chain. In other words, given the basic stabilizer chain $G^{(1)} = G \geq G^{(2)} \geq ... \geq G^{(n)} \geq G^{(n+1)} = 1$, let $S^{(i)} := G^{(i)} \cap S$ denote the strong generators which lie in $G^{(i)}$, and let $R^{(i)}$ be those relations in R which only involve elements of $S^{(i)}$. Then for a strong presentation, we have for each i that $G^{(i)} \cong \langle S^{(i)}|R^{(i)}\rangle$.

We will now show how, given a BSGS for a group, we can use it to perform constructive membership testing, in the form of an example.

Example 2.10. (Skip the example here and handwave the detail if running short on time - want no longer than 20 minutes elapsed by the start of the algorithm) Let $\Omega = \{1,2,3,4,5\}$, $H \leq \mathrm{Sym}(5)$, $H := \langle a := (1,2,3,4,5), b := (4,5,3,2)\rangle$. We ask whether $g = (1,2,5) \in H$. (In groups of this size we can fairly easily enumerate all the elements and check to see whether the given element is in the list, so pretend we're dealing with larger groups here).

A base for H is $[1,2]$. Indeed, $H_1 = \langle(4,5,3,2)\rangle$ and so $H_{1,2} = \mathrm{Id}$. This also shows that the generating set we have is actually a strong generating set for the base $\{1,2\}$. (Note that it wouldn't be a strong generating set for the base $\{2,1\}$, hence it would probably be more helpful to think of B as a sequence rather than a set).

Constructive membership testing works as follows; take the first base point 1, and compute $1^g = 2$. Check to see if 2 is in 1^H (otherwise g cannot possibly be in H). H is transitive, so this is clearly the case. In particular, we can find an element $h_1 \in H$ such that $1^{h_1} = 2$; for instance, take $h_1 =$

$a = (1, 2, 3, 4, 5)$. (There exist algorithms to do this efficiently, typically this falls out as a result of finding orbits of H). Produce a new element $t_1 := gh_1^{-1}$, so $t_1 = (1, 2, 5)(1, 5, 4, 3, 2) = (2, 4, 3)$. Then notice that by construction t_1 stabilizes 1. Do the same computation again with 2, only this time instead of H we consider $H_1 = \langle b = (4, 5, 3, 2)\rangle$. $2^{t_1} = 4$ and $2^{H_1} = \{2, 3, 4, 5\}$. Take $h_2 = b$ and set $t_2 = t_1 * h_2^1 = g * h_1^{-1} * h_2^{-1} = (4, 5)$. Notice again by construction that t_2 now stabilises 1 and 2. If our base were larger, we would iterate this procedure through all base points.

The point here is that now, if g were in H, then t_2 would be in $H_{1,2} = \mathrm{id}$. As $t_2 \neq \mathrm{id}$, g must not have been in H to start with. Conversely, if $|B| = m$ and $t_m = 1$, then we have $g * h_1^{-1} * ... * h_m^{-1} = 1$, so $g = h_m * ... * h_1$. Thus, not only is $g \in H$, but we have written it as a product of elements of H, each of which can easily be written in terms of the strong generators (indeed the computational method which produces these will typically produce such an element in precisely this manner), meaning we have written g (non-uniquely) as a product of the strong generators of H.

Another use of a BSGS of a group is that it can be informative about the order of the group; in particular it offers us a bound on the size of the groups (various levels of sophistication exist to get this bound). A very crude bound is obtained by noting that each element of H is determined uniquely by its action on the various base points; hence if we have a base B for a group G then we know that $|G| \leq |\Omega|^{|B|}$. For instance in the above example we can see that $|H| \leq 5^2 = 25$. As H is 2-transitive we have $|H| \geq 20$. From this and Lagrange we can conclude without further computation that $|H| = 20$. ($|H| \neq 24$ as it contains an element of order 5).

3 The algorithm

3.1 Setup

We introduce the notation that we will use throughout the description of this algorithm.

The VERIFY algorithm takes as input a proposed base $B = \{b_1, ..., b_m\}$ and strong generating set S for a group G, and produces a strong presentation for the group G, as well as verifying that the input is a BSGS, correcting it if not.

We construct a putative 'basic stabilizer chain' by defining $G^{(i)} := \langle S \cap G_{\{b_1,...,b_i\}}\rangle \leq G_{(b_1,...,b_i)}$; thus this is the group generated by those strong generators in S which stabilise the proposed base points; this will generate a subgroup of the stabilizer we seek at each point, and it will generate the required stabilizer iff S is a strong generating set, by definition.

We will describe one step of this algorithm - thus, assume that $K = G^{(i)}$ for some i, that $G_{(b_1,...,b_{i+1})} = G^{(i+1)}$ and that we have a strong presentation for K on its strong generators, $Y := S \cap K$. Let H denote the next step in the basis stabilizer chain, so $H = G^{(i-1)}$, and define $\alpha := \beta_i$. We seek to decide whether $H_\alpha = K$; if so, then $\{b_{i-1}, b_i, ..., b_m\}$ form a base for H with strong generating set $X := S \cap H$; otherwise we will need to extend either our strong generating set, so will want to find an element of $H_\alpha \setminus K$ to append to our strong generating set, or our base, so we will want to find an additional base point.

The calculation depends on the size of $|X \setminus Y|$. We will only consider the single generator case when $X \setminus Y = \{c\}$ consists of a single element; there are additional technicalities when this number is larger, but they generally boil down to multiple applications of the single generator case with a few additional computations.

3.2 The single generator case

The case where $X \setminus Y$ consists of a single generator c is the simplest to describe. We will simultaneously describe the algorithm and work through an example.

Example 3.1. Let $H \leq \mathrm{Sym}(10)$ be given by

$H = \langle a := (3,4)(5,8)(6,7)(9,10), b := (2,5,8)(3,7,9)(4,10,6), c := (1,2)(3,4)(5,6)(7,8)\rangle$. We will verify the proposed base $B = [1,2,3]$ and strong generating set $S = [a,b,c]$. This gives rise to the putative basic stabiliser chain $H^{(0)} = H \geq H^{(1)} = \langle a,b\rangle \geq H^{(2)} = \langle a\rangle \geq H^{(3)} = \langle \mathrm{id}\rangle$. Let $K = \langle a,b\rangle$, and assume that we have run VERIFY up to K, so we have confirmed that $H_{1,2} = \langle a\rangle$ and we have a presentation for K in terms of a and b. It turns out that $K \cong D_6$, and we will use the presentation $K \cong \langle x,y | x^2 = y^3 = (xy)^2 = 1\rangle$ (with x in the presentation corresponding to a in H and y corresponding to b. When we construct the full strong presentation of the group we will also use z as the element relating to c.)

We will use the algorithm to perform the final step of verification which decides if $K = H_{(1)}$, and subsequently produces a presentation for H.

Remark 3.2. We first set $\beta := \alpha^{c^{-1}}$ and $\Delta := \alpha^H$. We compute generators for K_β using MSTAB and choose orbit representatives of K_β on Δ; call these $[\gamma_1 := \alpha, \gamma_2 := \beta, \gamma_3, ..., \gamma_m]$. These will be used to compute the relations R_1.

Example 3.3. To lump all of our notation together, here is a list of what everything is in this example:

- $H = \langle a,b,c\rangle$ and $X = \{a,b,c\}$
- $K = \langle a,b\rangle$ and $Y = \{a,b\}$
- $c = X \setminus Y = c$ (notation has been carefully chosen here!)
- $\alpha = 1$

This is all in the setup of the algorithm, now we begin the algorithm itself. $c^{-1} = c$ so we have $\beta = 1^c = 2$. H is transitive so $\Delta = [1,2,3,4,5,6,7,8,9,10]$. Here we happen to know already that $K_2 = \langle a\rangle$; if we didn't, then we would do this by randomly generating elements of K_2 until we got a group of the right order (we know what order we want by Orbit-Stabilizer, since we know $|K|$ and it is easy to find β^K). We now produce the γ_i, which correspond to representatives of the orbits of a on Δ. We take $[\gamma_1, ..., \gamma_6] = [1,2,3,5,6,9]$.

Remark 3.4. We also choose orbit representatives $[\lambda_1 := \alpha.\lambda_2 := \beta, \lambda_3, ..., \lambda_l]$ of K on Δ, and simultaneously define transversals $\{\tau(\delta) : \delta \in \Delta\}$ of H_α in H. (By a *transversal* here, we mean a set of elements of H [stored as words in $\hat{X}$] such that for every δ, $\alpha^{\tau(\delta)} = \delta$.) We do this as follows in a specific way - this is for technical reasons to ensure the algorithm completes correctly. (Probably best illustrated in the next example rather than given explicitly).

We begin by setting $\lambda_1 := \alpha$ and $\lambda_2 := \beta$; simultaneously defining $\tau(\lambda_1) := \epsilon$ (the empty word) and $\tau(\lambda_2) := c^{-1}$, and proceed inductively. Upon defining λ_i we perform the following (dealing with the cases $i = 1$ and $i = 2$ together):

- Compute λ_i^K. For each $\lambda \in \lambda_i^K$, by definition we must have some $k \in K$, and thus corresponding word $\hat{k} \in \hat{Y}$, such that $\lambda_i^k = \lambda$; thus we set $\tau(\lambda) = \tau(\lambda_i)\hat{k}$.
- Check if $\cup_{j=1}^i \lambda_i^K = \Delta$. If so, then we are done; if not, then by definition of Δ we must have some $j \leq i$ and $\lambda \in \lambda_j^K$ such that $\lambda^c \notin \cup_{j=1}^i \lambda_i^K$. In this case, set $\lambda_{i+1} := \lambda^c$ and $\tau(\lambda_{i+1}) := \tau(\lambda)\hat{c}$.

Example 3.5. We follow the example. We first set $\lambda_1 = 1$ and $\lambda_2 = 2$. It is no surprise that $1^K = \{1\}$ given our choice of K to be generated by elements which stabilize 1. $2^K = \{2,5,8\}$. We then produce the transversals. $\tau(1) = \epsilon$, the empty word. $\tau(2) = c^{-1}$. We get from 2 to 5 in K by multiplying by b, so we set $\tau(5) = c^{-1}b$. Similarly $\tau(8) = c^{-1}b^{-1}$. (This is relatively arbitrary; we could equally take $c^{-1}ba$ and this will give us a different, but still accurate, presentation).

This evidently doesn't give us all of Δ so we need to define a new λ_i in the specific way described. In this case we see that $5 \in 2^K$ but $5^c = 6$ which we do not have yet, so we set $\lambda_3 = 6$. We discover that $6^K = \{6,7,4,10,9,3\}$ so we now cover all of Δ. We define the transversal by appending to the transversal for 6, which is given by $\tau(6) = c^{-1}bc$. (Here we must choose this transversal for the algorithm to work). For example, we get from 6 to 7 by applying a so we set $\tau(7) = c^{-1}bca$. Similarly we define the rest: $\tau(4) = c^{-1}bcb$, $\tau(10) = c^{-1}bcb^{-1}$, $\tau(9) = c^{-1}bcab$, $\tau(3) = c^{-1}bcab^{-1}$.

$$\begin{aligned}
\tau(1) &= \epsilon \\
\tau(2) &= c^{-1} \\
\tau(3) &= c^{-1}bcab^{-1} \\
\tau(4) &= c^{-1}bcb \\
\tau(5) &= c^{-1}b \\
\tau(6) &= c^{-1}bc \\
\tau(7) &= c^{-1}bca \\
\tau(8) &= c^{-1}b^{-1} \\
\tau(9) &= c^{-1}bcab \\
\tau(10) &= c^{-1}bcb^{-1}
\end{aligned}$$

Remark 3.6. We are now in a position to compute R_2 and R_3

For R_2, consider all elements of the form $\tau(\gamma_k)\hat{c}\tau(\gamma_k^c)^{-1}$. We have

$$\alpha^{\tau(\gamma_k)c\tau(\gamma_k^c)^{-1}} = \gamma_k^{z\tau(\gamma_k^c)^{-1}} = \alpha$$

so we must have that $\tau(\gamma_k)c\tau(\gamma_k^c)^{-1}$ represents an element of K if $K = H_\alpha$. Thus, we can test the element of H given by $\tau(\gamma_k)c\tau(\gamma_k^c)^{-1}$ for membership of K. If this element is not in K, then we know that $K \neq H_\alpha$ and we can return the element as a proposed addition to the BSGS. Otherwise, if the element is in K, we can write it as a word $\hat{u}$ in $\hat{Y}$, whereupon $\tau(\gamma_k)c\tau(\gamma_k^c)^{-1}\hat{u}^{-1}$ is a word which evaluates in H to 1_H and so we can add this to R_1.

Example 3.7. There are 6 computations to do here; I will only do informative ones explicitly (probably γ_1 and γ_6 only):

- $\tau(\gamma_1)c\tau(\gamma_1^c)^{-1}$. $\gamma_1 = 1$, so $\gamma_1^c = 2$ and this gives ϵcc. So we test c^2 for membership of K. $c^2 = \text{id} \in K$, so this gives us the relation z^2 to append to our presentation.

- (Optional) $\tau(\gamma_2)c\tau(\gamma_2^c)^{-1}$. $\gamma_2 = 2$ and $\gamma_2^c = 1$, giving $c^{-1}c\epsilon = \text{id}$. So this gives us no relation at all.

- $\tau(\gamma_6)c\tau(\gamma_6^c)^{-1}$. $\gamma_6 = 9$ and $\gamma_6^c = 9$. This gives $c^{-1}bcabcb^{-1}a^{-1}c^{-1}b^{-1}c$. Evaluating this directly we get $(2,8)(3,6)(4,9)(7,10)$. We can actually use the constructive membership testing algorithm we discussed earlier to test this for membership of K in exactly the same way, and we discover that this is actually $a^{-1}b^{-1}$. Hence this gives the relation $z^{-1}yzxyzy^{-1}x^{-1}z^{-1}y^{-1}zyx$.

- The others are similar: γ_4 gives us a trivial relation, γ_5 gives a relation $z^{-1}yz^2y^{-1}z$, which, using $z^2 = \text{id}$ easily reduces to a trivial relation. (The vanilla algorithm doesn't do any of these relations so often it will produce a presentation with a large number of redundant relations of this form. It is possible to reduce these computationally, but if all you want is a minimal strong presentation for the group then there exist faster algorithms than this one which do that for you). γ_3 gives another long relation $z^{-1}yzxy^{-1}zy^{-1}z^{-1}y^{-1}zxy$.

Hence $R_2 = \{z^2, z^{-1}yzxyzy^{-1}x^{-1}z^{-1}y^{-1}zyx, z^{-1}yzxy^{-1}zy^{-1}z^{-1}y^{-1}zxy\}$

Remark 3.8. To construct R_3, we consider each λ_i for $1 \leq i \leq l$. We compute generators $y_1, ..., y_t$ for K_{λ_i} (we have done this already for K_{λ_1}). We then express each y_i as a word in $\hat{Y}$. Then we have that

$$\alpha^{\tau(\lambda_i)\hat{y_j}\tau(\lambda_i)^{-1}} = \lambda_i^{\hat{y_j}\tau(\lambda_i)^{-1}} = \lambda_i^{\tau(\lambda_i)^{-1}} = \alpha$$

so as before we can test this element for membership of K. If it is not, then we return this element as a new proposed strong generator; otherwise, it lies in K and thus can be written as a word $\hat{u}$ in $\hat{Y}$; whereupon $\tau(\lambda_i)\hat{y_j}\tau(\lambda_i)^{-1}u^{-1}$ evaluates to 1_H and we can add this to R_2.

Example 3.9. $K_{\lambda_1} = K_1 = K = \langle a, b \rangle$, $K_{\lambda_2} = K_2 = \langle a \rangle$ and $K_{\lambda_3} = K_6 = \langle \text{id} \rangle$, so there is very little to check here. λ_1 simply asks if a and b are in K, which they clearly are, so this offers no interesting relations. We check $\tau(\lambda_2)a\tau(\lambda_2)^{-1} = c^{-1}ac$ for membership of K. Computing directly, $c^{-1}ac = a$, so clearly this is in K and this tells us that $c^{-1}aca^{-1} = \text{id}$. Hence we add the relation $z^{-1}xzx^{-1}$.

If the algorithm completes these steps without rejecting any elements as not belonging to K, then we have that $\langle\ \hat{X} \mid R \cup R_1 \cup R_2\ \rangle \cong H$. From the way we have constructed this it is clear that these relations are true in H; it is not obvious that these actually give the entire group we are interested in. The proof is not especially long but it is tedious, notation heavy and not particularly informative. See Derek's book if you're interested.

Example 3.10. This gives the full presentation of H as
$\langle x, y, z | x^2, y^3, xyxy, z^2, z^{-1}yzxyzy^{-1}x^{-1}z^{-1}y^{-1}zyx, z^{-1}yzxy^{-1}zy^{-1}z^{-1}y^{-1}zxy, z^{-1}xzx^{-1} \rangle$.

It turns out that the relations z^2 and $z^{-1}xzx^{-1}$ are superfluous and can be safely removed, but none of the others are; in particular both long relations are necessary in this presentation.

4 Additional material

??Make the screen slightly larger than normal, G is quite big.

```
load "C://MAGMA/GraduateSeminar.txt";
H; \\ H here is GL(3,3);
#H;
Hpres,B,S:=MPres(H);
B;
B[1];
Stabilizer(H,B[1]);
#Stabilizer(H,B[1]);
Stabilizer(H,B); \\ Shows that B is a base for H.
#S;
Hpres;
#Generators(Hpres);
Order(Hpres);
#H;

G;
#G;
Gpres,B,S:=MPres(G);
B;
S;
#S;
Gpres;
#Relations(Gpres);
```

This method, or rather a hybrid method combining this with a method of coset enumeration (see course on presentations of groups here) for the smaller groups in the basic stabilizer chain, is vastly more efficient than the MAGMA default Verify for large groups. For example, $\mathrm{SL}(7,5)$ is a group of order roughly 3×10^{33}. Given a random proposed base and strong generating set, I stopped the inbuild Verify after 14 hours on my laptop. Running the default Verify algorithm as described here completed in just under 3 hours; combining the two and using the inbuilt Verify for around the first half of the base points meant that it completed in a little over 10 minutes.