

**Mathematics 6310**  
**Computation in permutation groups**  
**Ken Brown, Cornell University, July 2011**

Let  $G$  be a group of permutations of a finite set. For definiteness, we can take  $G$  to be the group of permutations of  $\{1, 2, 3, 4, 5\}$  generated by the 4-cycles  $a := (1\ 2\ 4\ 3)$  and  $b := (1\ 2\ 5\ 4)$ . We will refer to this as our “running example” in what follows. What is the order of  $G$ ? Is  $G$  solvable? Is  $G$  nilpotent? What are the Sylow subgroups of  $G$ ? Can we find normal forms for the elements of  $G$  as words in the generators? Can we find a set of defining relations for  $G$ ? Computational group theorists have developed algorithms for answering questions like these. Most such algorithms require first constructing a “base” and “strong generating set” for the given permutation group. The purpose of this handout is to give a very brief introduction to these concepts. For more information see the books by Holt [1], Seress [2], or Sims [3].

As a warm-up, we begin with a fairly trivial algorithm.

1. ORBITS

Let  $G$  be a group with a finite generating set  $S$ . Suppose  $G$  acts on a set  $X$  on the right. [Throughout this handout I will use *right* group actions and will define composition in permutation groups accordingly. This makes it easier for you to draw Schreier graphs as you read, which I encourage you to do.] Given  $x \in X$ , how would you compute the  $G$ -orbit of  $x$ ? In our running example, you can see at a glance that the orbit of 1 (or any other element) is the entire set  $X := \{1, 2, 3, 4, 5\}$ . But if  $X$  is large, we need a more systematic procedure. Here’s a simple algorithm.

We will build the orbit as a list  $Y = \{y_1, \dots, y_k\}$  without repetition. Start with  $y_1 := x$ . Now run through the generators  $s \in S$  and adjoin  $y_1s$  to  $Y$  if it is not already present. [By “adjoin”, I mean set  $y_2 := y_1s$  for the first such  $s$ , then define  $y_3$ , and so on.] If nothing has been adjoined, stop. Otherwise, repeat the process with  $y_1$  replaced by  $y_2$ , i.e., successively adjoin  $y_2s$  to  $Y$  for each generator  $s$  such that  $y_2s$  is not already in  $Y$ . Continue in this way until you reach an index  $k$  such that  $Y$  already contains all  $y_ks$ ; this must happen because  $X$  is finite. Then  $Y$  is the orbit. Indeed,  $Y$  is obviously contained in the orbit and is  $G$ -invariant by construction, so it must be the whole orbit.

**Exercise 1.** Carry out the algorithm for our running example, with  $x = 1$ .

**Exercise 2.** The assertion that  $Y$  is  $G$ -invariant makes tacit use of the assumption that  $X$  is finite. Explain where this is used. How would you modify the procedure to work if  $X$  is infinite (still assuming, for simplicity, that  $G$  is finitely generated)?

Note that if we are given a finitely-generated group  $G$  and a subgroup  $H$  of finite index, we can apply the procedure above to the natural action of  $G$  on  $H \backslash G$ . Thus we have an algorithm for computing the index of  $H$  in  $G$ .

2. TRANSVERSALS

The orbit algorithm in the previous section not only computes the orbit  $Y = xG$ , but it leads to a specific way of representing each  $y \in Y$  as  $y = xg$  for some  $g = g(y) \in G$ . For when  $y_i$  is first adjoined to  $Y$ , it is given as  $y_j s$  for some  $j < i$  and  $s \in S$ ; so we can set  $g(y_i) := g(y_j)s$ . To start the induction, take  $g(y_1) := \text{id}$ .

Note that the elements  $g(y)$  form a right transversal for  $G_x$  in  $G$ , where  $G_x$  is the stabilizer of  $x$ . (A *right transversal* of a subgroup is a set of representatives for the right cosets.)

*Remark.* The construction of  $g(y)$  actually gives a specific  $S$ -word representing  $g(y)$ . In other words, we have constructed for each  $y \in Y$  a path  $\gamma(y)$  from  $x$  to  $y$  in the Schreier graph.

**Exercise 3.** Carry out the algorithm for our running example, with  $x = 1$ . Draw the Schreier graph, and indicate the paths  $\gamma(y)$ . (They should form a tree, with edges directed away from the root 1.)

Note that if we are given a finitely-generated group  $G$  and a subgroup  $H$  of finite index, we can apply the procedure above to the natural action of  $G$  on  $H \backslash G$ . Thus we have an algorithm for computing a right transversal for any subgroup of  $G$  of finite index.

### 3. STABILIZERS

Further analysis of the orbit algorithm leads to a computation of the stabilizer  $G_x$  of  $x$  in  $G$ . Suppose we are carrying out the algorithm and drawing the Schreier graph as we go along. At each step we have some  $y \in Y$  and  $s \in S$ , and we ask whether  $z := ys$  is already in the part of the graph that we have constructed. If not, we adjoin  $z$  and an edge  $e$  (labeled by  $s$ ) from  $y$  to  $z$ , and we have  $\gamma(z) = \gamma(y) * e$ , where  $*$  denotes path composition. Suppose, however, that  $z$  was already in the graph. Then we can still draw an edge  $e$  labeled by  $s$  from  $y$  to  $z$ , and we get a loop  $\gamma(y) * e * \overline{\gamma(z)}$  based at  $x$ , where the bar denotes path inversion. This loop represents an element

$$h(y, s) := g(y)sg(z)^{-1}$$

of the stabilizer  $G_x$ . The following result is due to Schreier:

**Theorem.** *The elements  $h(y, s)$  generate  $G_x$ .*

If  $k := |Y|$  and  $n := |S|$ , then the number of generators of  $G_x$  obtained in this way is

$$kn - (k - 1) = k(n - 1) + 1.$$

In practice, many of these generators are often redundant. In general, however, this upper bound on the number of generators of  $G_x$  is sharp. See the remark below.

**Exercise 4.** Explain where the number  $kn - (k - 1)$  above comes from.

**Exercise 5.** Carry out the algorithm for our running example, with  $x = 1$ . Eliminate redundant generators and give the simplest description you can of  $G_x$ .

**Exercise 6.** Prove Schreier's theorem. [Hint: Let  $H$  be the subgroup of  $G$  generated by the elements  $h(y, s)$ . Since  $H$  fixes  $x$ , there is a surjective  $G$ -set map  $H \backslash G \rightarrow Y$ . Construct a  $G$ -set map in the other direction.]

Note that if we are given a finitely-generated group  $G$  and a subgroup  $H$  of finite index, we can apply the procedure above to the natural action of  $G$  on  $H \backslash G$  to get a (finite) set of generators of  $H$ . If  $G$  has  $n$  generators and  $H$  has index  $k$ , then the number of generators of  $H$  obtained in this way is  $k(n - 1) + 1$ . If  $G$  is the free group on  $n$  generators, one can show that  $H$  is free on  $k(n - 1) + 1$  generators; so the upper bound  $k(n - 1) + 1$  on the number of generators required by  $H$  is sharp.

If you know a little algebraic topology, you might enjoy giving a topological proof of the assertion above about free groups. It is based on the following two facts: (1) If  $X$  is a connected graph, then its fundamental group is free of rank  $n$ , where  $n$  is defined by the equation

$$\chi(X) = 1 - n.$$

Here  $\chi(-)$  denotes the Euler characteristic. (2) If  $X$  is a finite complex and  $Y \rightarrow X$  is a  $k$ -fold covering map, then

$$\chi(Y) = k \cdot \chi(X).$$

#### 4. ORDER

If you have been doing the exercises involving our running example, you should know that the group in that example has order 20. [ $G$  acts transitively on a 5-element set, and the stabilizer of 1 has order 4.] The same idea can be used to compute the order of any group of permutations of a finite set  $X$ .

Let  $G$  be such a group, with a given set of generators. If  $G$  is trivial, its order is 1. Otherwise, choose an element  $x \in X$  that is not fixed by  $G$ . [It suffices to choose a nontrivial generator  $s$  and then take  $x$  that is not fixed by  $s$ .] Compute the  $G$ -orbit  $Y$  of  $x$  and the stabilizer  $H$  of  $x$ . Then  $H < G$ , and

$$|G| = |Y| \cdot |H|.$$

If  $H$  is trivial, we are done. Otherwise, repeat the process with  $G$  replaced by  $H$ . Continuing in this way, we obtain a sequence of elements  $x_1, \dots, x_k \in X$  and a strictly decreasing chain of subgroups

$$G = G^1 > G^2 > \dots > G^k > G^{k+1} = \{1\},$$

such that  $G^{i+1}$  is the stabilizer of  $x_i$  in  $G_i$  for  $i = 1, \dots, k$ . If  $Y_i$  is the  $G_i$ -orbit of  $x_i$ , then

$$|G| = |Y_1| \cdot |Y_2| \cdots |Y_k|.$$

In our running example, we can take  $x_1 = 1$  and  $x_2 = 2$  (so  $k = 2$ ). Then  $Y_1 = \{1, 2, 3, 4, 5\}$ ,  $G_2$  is generated by the 4-cycle  $(2354)$ , and  $Y_2 = \{2, 3, 4, 5\}$ . Hence

$$|G| = |Y_1| \cdot |Y_2| = 5 \cdot 4 = 20.$$

#### 5. BASES AND STRONG GENERATING SETS

The set  $\{x_1, \dots, x_k\}$  constructed in the previous section has the property that nothing in  $G$  except the identity fixes all  $x_i$ . Such a set is called a *base* for the permutation group  $G$ . Any base gives rise to a chain of subgroups

$$G = G^1 \geq G^2 \geq \dots \geq G^k \geq G^{k+1} = \{1\},$$

where  $G^{i+1}$  is the stabilizer of  $x_i$  in  $G_i$  for  $i = 1, \dots, k$ . And any generating set for  $G$  that includes generators for each  $G^i$  is called a *strong generating set* for  $G$ .

The inductive procedure described above for finding a base yields a generating set for each  $G_i$ , so we can simply take the union of these to get a strong generating set. That procedure, however, is very inefficient because the sizes of the generating sets (obtained via Schreier's theorem) increase very rapidly. There are better algorithms in which, roughly speaking, one tries to construct the chain  $(G^i)$  all at once. At each step one has elements  $x_1, \dots, x_l$  and a chain

$$G = H^1 \geq H^2 \geq \dots \geq H^l \geq H^{l+1} = \{1\}$$

such that  $H^{i+1}$  stabilizes  $x_i$  but is not necessarily the full stabilizer of  $x_i$  in  $H^i$ . One then enlarges one or more of the  $H^{i+1}$  by adding new generators. (If  $i = l$ , one also has to enlarge the partial base by choosing a suitable  $x_{l+1}$ .) See the references for details.

## 6. TESTING FOR MEMBERSHIP

I mentioned at the beginning of this handout that most algorithms for permutation groups make use of a base and strong generating set (or BSGS). We have already seen how to use a BSGS to compute the order of the group. As another example, we indicate here how to test an arbitrary permutation of  $X$  to see if it is in our given group  $G$ . Assume that we have computed the orbit  $Y_i$  for each  $i$  and, as in Section 2, specific elements of  $G_i$  that map  $x_i$  to the elements of  $Y_i$  (i.e., a right transversal for  $G_{i+1}$  in  $G_i$ ). This is usually done in the course of finding the BSGS.

Let  $g$  be a permutation of  $X$ . Start by computing the image  $x_1g$ . If it is not in  $Y^1$ , then we know  $g \notin G^1 = G$  and we are done. Otherwise, we can find  $g_1 \in G$  such that  $x_1g = x_1g_1$ . Then  $g' := gg_1^{-1}$  fixes  $x_1$ , and  $g \in G \iff g' \in G$ , in which case  $g' \in G^2$ . Now check whether  $x_2g'$  is in  $Y_2$ . If not, then  $g' \notin G^2$ , and hence  $g \notin G$ . Otherwise, choose  $g_2 \in G^2$  such that  $x_2g' = x_2g_2$ . Continuing in this way, either we discover that  $g \notin G$  or we find an expression  $g = g_k \cdots g_2g_1$ , where each  $g_i$  is in our transversal for  $G_{i+1}$  in  $G_i$ .

**Exercise 7.** Let  $G$  be the group of our running example. Show that the transposition  $(1\ 2)$  is not in  $G$ .

## REFERENCES

- [1] Derek F. Holt, Bettina Eick, and Eamonn A. O'Brien. *Handbook of computational group theory*. Discrete Mathematics and its Applications (Boca Raton). Chapman & Hall/CRC, Boca Raton, FL, 2005, pp. xvi+514.
- [2] Ákos Seress. *Permutation group algorithms*. Vol. 152. Cambridge Tracts in Mathematics. Cambridge: Cambridge University Press, 2003, pp. x+264.
- [3] Charles C. Sims. *Computation with finitely presented groups*. Vol. 48. Encyclopedia of Mathematics and its Applications. Cambridge: Cambridge University Press, 1994, pp. xiii+604.