

The Schreier-Sims algorithm for finite permutation groups

Derek Holt

University of Warwick

First CoDiMa Training School in Computational Discrete
Mathematics,
Manchester, 16-20 November 2015

Contents

- 1 Permutation groups
- 2 Orbits and stabilizers
- 3 Computing orbits
- 4 Computing stabilizers
- 5 Bases and strong generating sets
- 6 Testing membership in a group with a BSGS
- 7 The Schreier-Sims Algorithm
- 8 Complexity
- 9 Improvements and further computations
- 10 Applications

Permutation groups

Definition

A **permutation group** is a subgroup of the group $\text{Sym}(\Omega)$ of all permutations of a set Ω .

In this talk, Ω will be finite, and can be assumed equal to $\{1, 2, \dots, n\}$. We write S_n for $\text{Sym}(\Omega)$.

For $\alpha \in \Omega$ and $g \in \text{Sym}(\Omega)$, the image of α under G will be written as α^g (and not $g(\alpha)$).

So, for $g, h \in \text{Sym}(\Omega)$, gh means first g , then h .

Example

$$n = 6, \quad g = (1, 3)(2, 5), \quad h = (1, 3, 5)(2, 6) \Rightarrow \\ gh = (1, 5, 6, 2), \quad hg = (2, 6, 5, 3).$$

Orbits and stabilizers

Let G be a permutation group on Ω .

For $\alpha \in \Omega$ we define the **orbit** and **stabilizer** of α under G by

$$\begin{aligned}\text{Orb}_G(\alpha) &= \alpha^G = \{\alpha^g : g \in G\} \\ \text{Stab}_G(\alpha) &= G_\alpha = \{g \in G : \alpha^g = \alpha\}.\end{aligned}$$

The distinct orbits partition Ω .

If $\beta \in \alpha^G$, then there exists $g \in G$ with $\alpha^g = \beta$. Then, for $h \in G$,

$$\alpha^h = \beta \Leftrightarrow \alpha^h = \alpha^g \Leftrightarrow hg^{-1} \in G_\alpha \Leftrightarrow h \in G_\alpha g$$

So α has the same image under g and h if and only if g and h are in the same right coset of G_α in G .

The Orbit-Stabilizer Theorem

Hence there is a bijection between α^G and the right cosets of G_α in G and:

Theorem (The Orbit-Stabilizer Theorem)

If G acts on Ω and $\alpha \in \Omega$, then $|\alpha^G| = |G|/|G_\alpha|$.

For a group $G \leq \text{Sym}(\Omega)$ generated by a finite set X of permutations, the following simple procedure computes the orbit α^G and, for each $\gamma \in \alpha^G$, it computes an element $u_\gamma \in G$ with $\alpha^{u_\gamma} = \gamma$.

These are returned as a sequence OT of pairs (γ, u_γ) .

Algorithm

ORBITTRANSVERSAL(α, X)**Input:** $\alpha \in \Omega$, $X = [x_1, \dots, x_r]$, $x_i \in \text{Sym}(\Omega)$ with $\langle X \rangle = G$ **Output:** $\{(\gamma, u_\gamma) : \gamma \in \alpha^G, \alpha^{u_\gamma} = \gamma\}$

- 1 OT := $[(\alpha, 1_G)]$;
- 2 **for** $(\gamma, u_\gamma) \in \text{OT}$, $x \in X$ **do if** $\gamma^x \notin \text{OT}$
- 3 **then** APPEND($\sim \text{OT}$, $(\gamma^x, u_\gamma x)$);
- 4 **return** OT;

Note that, by the above arguments,

$$G = \bigcup_{\gamma \in \alpha^G} G_\alpha u_\gamma.$$

In other words, the set $\{u_\gamma : \gamma \in \alpha^G\}$ is a **right transversal** of G_α in G .

To compute (generators for) the stabilizer G_α of α , we use the following result of Schreier.

Theorem

Let $G = \langle X \rangle$ be a group generated by a set X , let H be a subgroup of G , and let U be a right transversal of H in G . For any $g \in G$, we write \bar{g} for the unique element $u \in U$ with $g \in Hu$. Then H is generated by the set Y , with

$$Y = \{u\overline{ux}^{-1} : u \in U, x \in X\}.$$

Proof of Schreier's Theorem

Lemma

$$Y^{-1} = S, \text{ where } S = \left\{ ux^{-1}\overline{ux^{-1}}^{-1} : u \in U, x \in X \right\}.$$

Proof

Let $g \in Y^{-1}$, so $g = (ux\overline{ux^{-1}})^{-1} = \overline{ux^{-1}}x^{-1}u^{-1}$.

Let $v = \overline{ux} \in U$.

Then $\overline{vx^{-1}} = u$, so $g = vx^{-1}\overline{vx^{-1}}^{-1} \in S$.

Hence $Y^{-1} \subseteq S$.

Conversely, let $g = ux^{-1}\overline{ux^{-1}}^{-1} \in S$, so $g^{-1} = \overline{ux^{-1}}xu^{-1}$.

Let $v = \overline{ux^{-1}}$.

Then $\overline{vx} = u$, so $g^{-1} = vx\overline{vx}^{-1} \in Y$, so $g \in Y^{-1}$.

Hence $S \subseteq Y^{-1}$, so $Y^{-1} = S$.

Proof of Schreier's Theorem (ctd)

Proof of Theorem

Let $h \in H$. So $h = a_1 \cdots a_k$, with each $a_i \in A = X \cup X^{-1}$.

Define $u_0 := 1$, and for $1 \leq i \leq k$, $u_i := \overline{a_1 \cdots a_i}$.

Since $h \in H$, we have $u_k = \bar{h} = 1$. Then

$$h = (u_0 a_1 u_1^{-1}) (u_1 a_2 u_2^{-1}) \cdots (u_{i-1} a_i u_i^{-1}) \cdots (u_{k-1} a_k u_k^{-1}).$$

Note that $u_{i+1} = \overline{a_1 \cdots a_{i+1}}$ is in the same coset of H as $u_i a_{i+1}$, so $\overline{u_i a_{i+1}} = u_{i+1}$, and

$$h = (u_0 a_1 \overline{u_0 a_1}^{-1}) \cdots (u_{i-1} a_i \overline{u_{i-1} a_i}^{-1}) \cdots (u_{k-1} a_k \overline{u_{k-1} a_k}^{-1}).$$

Each bracketed term is in Y if $a_i \in X$, and in Y^{-1} if $a_i \in X^{-1}$ by the lemma, so $H = \langle Y \rangle$.

Using this theorem, the following procedure computes the list OT, and at the same time computes a set Y of generators for G_α .

Algorithm

ORBITTRANSVERSALSTABILIZER(α, X)

Input: $\alpha \in \Omega$, $X = [x_1, \dots, x_r]$, $x_i \in \text{Sym}(\Omega)$ with $\langle X \rangle = G$

Output: OT, Y , as described above

- 1 OT := $[(\alpha, 1_G)]$;
- 2 $Y := \{\}$;
- 3 **for** $(\gamma, u_\gamma) \in \text{OT}$, $x \in X$ **do if** $\gamma^x \notin \text{OT}$
- 4 **then** APPEND($\sim \text{OT}$, $(\gamma^x, u_\gamma x)$);
- 5 **else** INCLUDE($\sim Y$, $u_\gamma x (u_\gamma x)^{-1}$);
- 6 **return** OT, Y ;

Example

$G = \langle a, b \rangle \leq S_4$ with $a = (1, 2, 3)$, $b = (1, 4)$, $\alpha = 1$.

γ	u_γ	x	γ^x	u_{γ^x}	$y = u_\gamma u_{\gamma^x}^{-1}$
1	1	a	2	a	
1	1	b	4	b	
2	a	a	3	a^2	
2	a	b	2		$aba^{-1} = (3, 4)$
3	a^2	a	1		$a^3 = 1$
3	a^2	b	3		$a^2 ba^{-2} = (2, 4)$
4	b	a	4		$bab^{-1} = (2, 3, 4)$
4	b	b	1		$b^2 = 1$

So $G_\alpha = \langle (3, 4), (2, 4), (2, 3, 4) \rangle$.

But note that the third generator of G_α is redundant.

Bases and strong generating sets: definitions

From on, let $G \leq \text{Sym}(\Omega)$ with $\Omega = \{1 \dots n\}$, where G is generated by a finite sequence S of elements of $\text{Sym}(n)$.

Let $B = [\beta_1, \dots, \beta_k]$ be a sequence of distinct elements of Ω .

For $1 \leq i \leq k + 1$ define

$$G^{(i)} = G_{\beta_1, \dots, \beta_{i-1}} \quad (\text{so } G^{(1)} = G)$$

$$S^{(i)} = S \cap G^{(i)}$$

$$H^{(i)} = \langle S^{(i)} \rangle \quad (\Rightarrow H^{(i)} \leq G^{(i)})$$

$$\Delta^{(i)} = \beta_i^{H^{(i)}}$$

$$U^{(i)} = \{u_\gamma^{(i)} \mid \gamma \in \Delta^{(i)}\}$$

For given B and S , it is straightforward to compute $S^{(i)}$, and then $\Delta^{(i)}$ and $U^{(i)}$ can be computed by calling `ORBITTRANSVERSAL`($\beta_i, S^{(i)}$).

Definition

The sequence B is said to be a **base** for G if the only element in G that fixes each of β_1, \dots, β_k is the identity. That is, if $G^{(k+1)} = 1$, and hence

$$1 = G^{(k+1)} \leq G^{(k)} \leq \dots \leq G^{(2)} \leq G^{(1)} = G.$$

The sequence S is said to be a **strong generating set** for G relative to the base B if it includes generators for each stabilizer $G^{(i)}$ in the chain above; that is, for $i = 1, 2, \dots, k+1$, $G^{(i)} = \langle S^{(i)} \rangle = H^{(i)}$.

Note that this is true by definition for $i = 1$.

If B is a base and S is a strong generating set relative to B , then $G^{(i)}$ is called the i -th **basic stabilizer**, $\Delta^{(i)} = \beta_i^{G^{(i)}}$ the i -th **basic orbit**.

The normal form from a BSGS

If (B, S) is a BSGS, then the sequence of transversals $U^{(1)}, \dots, U^{(k)}$ provides us with a convenient normal form for the elements of G , since every element $g \in G$ has a unique representation

$$g = u_{\gamma_k}^{(k)} \cdots u_{\gamma_2}^{(2)} u_{\gamma_1}^{(1)}$$

with $\gamma_i \in \Delta^{(i)}$.

Then the order of the group can be read off from the transversals:

$$|G| = |U^{(k)}| \cdot |U^{(k-1)}| \cdots |U^{(1)}|.$$

For a given BSGS (B, S) , the next function **STRIP** tests whether a given $g \in S_n$ lies in G and, if so, calculates its normal form $[u_{\gamma_i}^{(i)}]$.

Algorithm

STRIP($g, B, S, [\Delta^{(i)}]$)

Input: $g \in \text{Sym}(\Omega)$, and BSGS $B, S, [\Delta^{(i)}]$ as described above

Output: Normal form $U = [u_{\gamma_i}^{(i)}]$ and $h \in \text{Sym}(\Omega)$
where $g \in G \Leftrightarrow h = 1$.

```
1   $h := g; U := [];$ 
2  for  $i \in [1..k]$ 
3      do (*  $h$  fixes base points  $\beta_1, \dots, \beta_{i-1}$  *)
4           $\gamma_i := \beta_i^h;$ 
5          if  $\gamma_i \notin \Delta^{(i)}$  then break;
6           $x := u_{\gamma_i}^{(i)};$ 
7          APPEND( $\sim U, x$ );
8           $h := hx^{-1};$ 
9  return  $U, h;$ 
```

The Schreier-Sims Algorithm

To simplify the description, we present a recursive version of the Schreier-Sims Algorithm, which tests whether (B, S) is a BSGS for G .

Algorithm

SCHREIERSIMS($B, S, [\Delta^{(i)}]$)

Input: Possible BSGS (B, S) with basic orbits $[\Delta^{(i)}]$

Output true if (B, S) is a BSGS for G , false if not.

```
1 if not SCHREIERSIMS( $B \setminus \{\beta_1\}, S \cap G_{\beta_1}, [\Delta^{(i)} : i > 1]$ )
2   then return false;
3  $\Delta, Y := \text{ORBITSTABILIZER}(\beta_1, S);$ 
4 for  $y \in Y$ 
5   do  $U, h := \text{STRIP}(y, B \setminus \{\beta_1\}, S \cap G_{\beta_1}, [\Delta^{(i)} : i > 1]);$ 
6     if  $h \neq 1$  then return false;
7 return true;
```


The Schreier-Sims Algorithm (ctd)

A practical implementation would differ from the above in several ways.

- (i) We would calculate the basic orbits during the algorithm rather than in advance.
- (ii) We would generate the elements y output by **ORBITSTABILIZER** one at a time and apply **STRIP** to each one in turn.
- (iii) On failure, we would adjoin the 'failing' element h to S , and possibly a new point to B , and then resume the computation, avoiding recalculating orbits that have not changed.

To avoid repeated failures, it is helpful to construct a probable BSGS before applying **SCHREIERSIMS**. This can be done by applying **STRIP** to random elements of G , and stopping when we have chosen some number (say 20) of random elements without changing B and S .

If (B, S) is not a BSGS then the probability of detecting this by stripping a single random group element is at least $1/2$.

Example ($n = 5$)

$$S = S^{(1)} = [(1, 2, 3), (3, 4, 5)],$$

$$B = [1, 3],$$

$$S^{(2)} = [(3, 4, 5)],$$

$$\Delta^{(1)} = [1, 2, 3, 4, 5],$$

$$\Delta^{(2)} = [3, 4, 5].$$

ORBITSTABILIZER with

$$\gamma = 2, \quad x = (3, 4, 5), \quad \gamma^x = 2, \quad u_\gamma = u_{\gamma^x} = (1, 2, 3),$$

gives

$$y := u_\gamma x (u_{\gamma^x})^{-1} = (2, 4, 5) \in Y. \quad \text{Then}$$

STRIP($y, [3], [(3, 4, 5)], [\Delta^{(2)}]$) fails, and so does

SCHREIERSIMS($B, S, [\Delta^{(i)}]$).

Example (ctd)

We append y to S , and 2 to B , giving

$$S = S^{(1)} = [(1, 2, 3), (3, 4, 5), (2, 4, 5)],$$

$$B = [1, 3, 2],$$

$$S^{(2)} = [(3, 4, 5), (2, 4, 5)],$$

$$S^{(3)} = [(2, 4, 5)],$$

$$\Delta^{(1)} = [1, 2, 3, 4, 5],$$

$$\Delta^{(2)} = [3, 4, 5, 2],$$

$$\Delta^{(3)} = [2, 4, 5].$$

SCHREIERSIMS($B, S, [\Delta^{(i)}]$) now succeeds.

Complexity

A straightforward implementation of **SCHREIERSIMS** runs in polynomial time, but $\Theta(n^5)$ is the worst case. It is very effective for small to moderate n (say $n \leq 10^3$ or 10^4 , depending on $|B|$).

There are very fast randomised versions in which we use **ORBITSTABILIZER** to generate random elements of the set Y of Schreier generators, and test those using **STRIP**, and stop after a certain number (say 20) have passed the test consecutively. Of course this could return a wrong answer, with inconsistent data.

If we know the order of the group G already, then we can use the random method very effectively by stopping when $\prod_{i=1}^k |\Delta^{(i)}| = |G|$.

If we know already that B is a base (for example when G is given as a subgroup of a group known to have base B) then to test that $h = 1$, we only need check that h fixes the points in B . This produces a significant speed-up if $k = |B|$ is small compared with n .

Improvements and further computations

There are many possible improvements, some of which incorporate the randomised algorithm but with more elaborate methods to check correctness. One such method uses another fundamental algorithm in Computational Group Theory, namely **coset enumeration**.

The BSGS data structure is used in almost all further algorithms for structural computations in finite permutation groups.

The normal form $[u_{\gamma_i}^{(i)}]$ provides a convenient representation for **backtrack searches** through the group elements.

These are used in computing **Sylow subgroups, centralizers** and **normalizers**, for example.

More specialised versions of **SCHREIERSIMS** were used by Sims to construct and prove the existence of some of the sporadic finite simple groups, including

- (i) the **Lyons group** in 1973 ($n \approx 9 \times 10^6$);
- (ii) the **O’Nan group** in 1976 ($n = 122760$); and
- (iii) the **Baby Monster** ($n \approx 1.4 \times 10^{10}$) in 1980.

An implementation of **SCHREIERSIMS** was used in the early 1980s by Diaconis, Graham, Kantor, and Knuth to help determine the group $\text{char}(2n)$ generated by the two types of perfect riffle shuffles in decks with an even number $2n$ of cards.

For example, $\text{char}(12) \cong C_2^{11} \rtimes M_{12}$.

It has also been used to construct certain Cayley graphs, which can be used in the design of processor interconnection networks.