Chapter 8. Regularity

This chapter is concerned with the question: Is there a permutation (other than the identity) in the group which fixes some point? We wish to answer this question using just the generators of the group. We do not want to look at the individual elements of the group.

The algorithms of this chapter illustrate the concepts of orbits and Schreier vectors. In particular, the use made of Schreier vectors is a typical one. It is hoped that this chapter will reinforce the reader's understanding of these important concepts.

Definitions and a First Algorithm

A permutation group $G = \langle s_1, s_2, \cdots s_m \rangle$ is *regular* if Ω is an orbit of G and there is no element of G (other than the identity) that fixes a point.

For example, the group of degree 8 generated by a=(1,2,3,4)(5,6,7,8) and b=(1,5)(2,8)(3,7)(4,6) is regular. It is not difficult to see that Ω is an orbit. The group has precisely eight elements,

identity
(1,2,3,4)(5,6,7,8)
(1,3)(2,4)(5,7)(6,8)
(1,4,3,2)(5,8,7,6)
(1,5)(2,8)(3,7)(4,6)
(1,8)(2,7)(3,6)(4,5)
(1,7)(2,6)(3,5)(4,8)
(1,6)(2,5)(3,8)(4,7)

none of which (other than the identity) fixes any point.

A permutation group is *semiregular* if there is no element (other than the identity) that fixes a point.

The group of degree 8 generated by a=(1,2)(3,4)(5,6)(7,8) and b=(1,3)(2,4)(5,7)(6,8) is semiregular. The group has precisely four elements

identity
a = (1,2)(3,4)(5,6)(7,8)
b = (1,3)(2,4)(5,7)(6,8)
$a \times b$ = (1,4)(2,3)(5,8)(6,7)

none of which (other than the identity) fixes any point. The group is not regular because its orbits are {1,2,3,4} and {5,6,7,8}.

The theoretical fact on which we base our algorithm is

Lemma

Let α be any point in Ω. If Ω is an orbit of G, and if, for each generator s of G there is a permutation z of Ω such that $\alpha^z = \alpha^s$ and such that z commutes with all the generators of G, then G is regular.

Suppose α is the representative of the orbit Ω of G, and that v is a Schreier vector of the orbit with respect to the generators S. Since the permutation z of the lemma should commute with each generator, it should therefore commute with each element of the group. In particular, z should commute with *trace* (β, v), for all points β. Hence,

$$\beta^z = (\alpha^{trace(\beta,\, v)})^z = \alpha^{trace(\beta,\, v) \times z} = \alpha^{z \times trace(\beta,\, v)} = (\alpha^z)^{trace(\beta,\, v)}.$$

However, the other property of z is that we know α^z. Hence, the above equation determines the image of every point under the action of z. It remains to verify that the action defines a permutation, and that the permutation commutes with all the generators of G. The algorithm is presented as Algorithm 1.

Algorithm 1 : Test regularity

Input : a set S of generators of a permutation group G on Ω;

Output : whether G is regular or not;

begin

form the orbits and Schreier vector v of G;
if Ω is not an orbit **then exit** with result "not regular"; **end if**;

α := orbit representative;
for each generator s of G **do** (* form z such that $\alpha^z = \alpha^s$ *)
  $\gamma := \alpha^s$;
  for each point β in Ω **do**
    image := $\gamma^{trace(\beta,\, v)}$; $z[\beta]$:= *image*
  end for;
  if (z is not a permutation)
      or (z does not commute with all generators) **then**
    exit with result "not regular";
  end if;
end for;

exit with result "regular";
end;

We may as well check that z is a permutation as we construct it. To do this we check whether each point is used as an image precisely once. The modified algorithm is Algorithm 2.

Algorithm 2 : Test regularity

Input : a set S of generators of a permutation group G on Ω;

Output : whether G is regular or not;

```
begin

  form the orbits and Schreier vector v of G;
  if Ω is not an orbit then exit with result "not regular"; end if;

  α := orbit representative;
  for each generator s of G do (* form z such that α^z = α^s *)

    γ := α^s;
    for each point β in Ω do used[β] := false; end for;
    for each point β in Ω do
      image := γ^trace(β, v);
      if used[image] then
        exit with result "not regular";
      else
        z[β] := image;  used[image] := true;
      end if;
    end for;

    if z does not commute with all generators then
      exit with result "not regular";
    end if;

  end for;

  exit with result "regular";
end;
```

The main components of the cost of Algorithm 2 are the $|\Omega| \times |S|$ calls to *trace*, and checking commutativity. The call to *trace* does not need to return an element of the group. All we are interested in is the action of the element on the point γ. Thus *trace* can be modified, and the worst case cost of the modified algorithm is $3 \times |\Omega|$. To check if two permutations commute, we must check for each point that the image under their composition is the same irrespective of the order of composition. This requires $4 \times |\Omega|$ operations. Thus to check the commutativity of each z with each generators costs $4 \times |S|^2 \times |\Omega|$ operations.

The total cost of Algorithm 2 is bounded by

$$4 \times |\Omega|^2 \times |S| \;+\; 4 \times |\Omega| \times |S|^2 \;+\; 5 \times |\Omega| \times |S| \;+\; 6 \times |\Omega| \;+\; |S|$$

operations.

Parallel Testing of Regularity

We wish to eliminate the $|\Omega|^2 \times |S|$ term in the cost of the algorithm. This term is due to the calls to *trace* where the spanning tree may be of height $|\Omega|$. It is possible to combine the formation of the orbit and Schreier vector with the formation of all the z's, in effect reducing the cost of a call to *trace* to one operation. Of course, we need to store all the z's as they are being constructed, so our time saving comes at the cost of more space. This is presented as Algorithm 3.

Algorithm 3 : Parallel testing of regularity

```
Input : a set S of generators of a permutation group G on Ω;

Output : whether G is regular or not;

begin
  α := 1; orbit := {α};   (* α is orbit representative*)

  (*initialize information concerned with z's*)
  for each generator s of G do
    for each point β in Ω do used[s][β] := false; end for;
    z[s][α] := α^s;   used[s][α^s] := true;
  end for;

  (*construct orbit and z's at the same time*)
  for each point γ in orbit do
    for each generator g of G do

      if γ^g not in orbit then (*extend orbit*)
        add γ^g to orbit;

        (*determine one more image in each z*)
        for each generator s of G do
          image := z[s][γ]^g;
          if used[s][image] then
            exit with result "not regular";
          else
            z[s][γ^g] := image;   used[s][image] := true;
          end if;
        end for;

      end if;

    end for;
  end for;

  if orbit ≠ Ω then exit with result "not regular"; end if;
```

```
    for each generator s of G do
      if z[s] does not commute with each generator then
        exit with result "not regular";
      end if;
    end for;

    exit with result "regular";
  end;
```

Consider the group of degree 8 generated by a=(1,2,3,4)(5,6,7,8) and b=(1,5)(2,8)(3,7)(4,6). Algorithm 3 determines $z[a]$ = (1,2,3,4)(5,8,7,6) and $z[b]$ = (1,5)(2,6)(3,7)(4,8), both of which commute with the generators a and b. Hence, the group is regular.

Consider the group of degree 4 generated by a=(1,2,3,4) and b=(1,4)(2,3). Algorithm 3 determines $z[a]$=(1,2,3,4) and $z[b]$ = (1,4,3,2). They are both permutations and both commute with the generator a, but neither of them commutes with the generator b. Hence, the group is not regular.

The analysis is straightforward. Algorithm 3 is an $O(|\Omega| \times |S|^2)$ algorithm.

Testing Semiregularity

Suppose that the group G is not transitive. Then G has orbits

$$\Delta_0, \Delta_1, \cdots \Delta_{m-1}$$

with orbit representatives

$$\delta_0, \delta_1, \cdots \delta_{m-1}.$$

If G is semiregular then

(1) the orbits $\Delta_0, \Delta_1, \cdots \Delta_{m-1}$ all have the same size,

(2) G acting on the orbit Δ_0 is regular, **and**

(3) there is a permutation z of Ω such that

$$\delta_0^z = \delta_1, \delta_1^z = \delta_2, \cdots \delta_{m-1}^z = \delta_0,$$

and such that z commutes with all the generators of G.

The previous algorithm determines the truth of (2), so we will only consider determining the truth of (3). Again, the conditions uniquely determine the action of z. It remains to check that z is a permutation and that z does indeed commute with all the generators. The algorithm is Algorithm 4.

Algorithm 4 : Testing semiregularity

Input : a set S of generators of a nontransitive group G acting on Ω;

Output : whether G is semiregular;

begin

form the orbit representatives $\delta_0, \delta_1, ..., \delta_{m-1}$
  and a Schreier vector v of the orbit Δ_0;

use Algorithm 3 to decide whether G is regular on Δ_0;
if G is not regular on Δ_0 **then**
  exit with result "not semiregular";
end if;

(*construct z and check it is a permutation*)
for each point β in Ω **do** *used* [β] := false; **end for**;

for each point β in Δ_0 **do**
  g := *trace* (β, v);
  for i := 0 to $m-1$ **do**
    image := $\delta^g_{i+1 \ mod \ m}$;
    if *used* [*image*] **then**
      exit with result "not semiregular";
    else
      used [*image*] := true; $z\,[\delta_i^g]$:= image;
    end if;
  end for;
end for;

if z commutes with each generator of G **then**
  exit with result "semiregular";
else
  exit with result "not semiregular";
end if;

end;

Of course, we can determine z in parallel with constructing the orbit, and therefore effectively reduce the cost of the call to *trace* to one operation.

Consider the group of degree 8 generated by a=(1,2)(3,4)(5,6)(7,8) and b=(1,3)(2,4)(5,7)(6,8). The orbit representatives are 1 and 5. The group acting on Δ_0={1,2,3,4} is regular. The algorithm constructs z=(1,5)(2,6)(3,7)(4,8) which is a permutation and which commutes with both generators. Hence, the group is semiregular.

Summary

This chapter has presented efficient algorithms for deciding whether or not a permutation group, given by a set of generators, is regular or semiregular.

Exercises

(1/Easy) Write the modified version of *trace* mentioned in the analysis of Algorithm 2 and show that the cost of *trace* (β, v) is $3 \times d(\beta)$.

(2/Moderate) Modify Algorithm 4 to form z in parallel with constructing the orbit Δ_0, so that the effective cost of *trace* is one operation. Analyse the resulting algorithm.

Bibliographical Remarks

The algorithms to test regularity (and semiregularity) were described by Charles Sims in lectures given at Oxford in January and February 1973. The fundamental lemma can be proved using Proposition 4.3 in the book H. Wielandt, **Finite Permutation Groups**, Academic Press, New York, 1964.