Chapter 16. Sylow Subgroups

This chapter presents an algorithm for computing a Sylow p-subgroup of a permutation group. The algorithm uses particular homomorphisms in a divide-and-conquer fashion to reduce the computation to simpler groups.

Homomorphic Images of Centralizers

The methods that rely on homomorphisms to reduce the problem must address the issue of solving the problem (or a related one) in the image and kernel of the homomorphism, and to somehow combine those two solutions. The aim is to choose the homomorphism so that these tasks are easy. Generally, the combination will involve taking the preimage of the solution for the image. The preimage will contain the kernel if we are dealing with subgroups, or the preimage will be a coset of the kernel if we are dealing with an element. Therefore it is important that the kernel be very closely related to the final solution. In the problem of this chapter we wish to compute a Sylow subgroup. This is a subgroup of order p^m, where p^m is the largest power of the prime p dividing the order of G. If $f : G \rightarrow H$ is a homomorphism, and S is a Sylow p-subgroup of $f(G)$, then the preimage $f^{-1}(S)$ always contains a Sylow subgroup of G. The preimage will be a Sylow subgroup if and only if the kernel of f is a p-group.

The first approach we consider uses a homomorphism that has a p-group as its kernel. However, we cannot guarantee that such homomorphisms exist when the domain is G, but we can if we restrict the domain to be a subgroup of G. Of course, we want the subgroup to contain a Sylow p-subgroup of G. The subgroup we use as the domain is a centralizer C in G of an element z of order p.

Let P be a Sylow p-subgroup of G. The centre $Z(P)$ of P is the subgroup of all elements of P which commute with every element of P. As P is a p-group, it has a non-trivial centre, and if z is any element in $Z(P)$ then $C_G(z)$ contains P. We can always take z to have order p (by taking a suitable power of the element). So we can always find a suitable element z and compute its centralizer C. The elements of C permute the cycles of z so the partition π of Ω given by the cycles of z is invariant under C and determine a blocks homomorphism. The elements in the kernel of this homomorphism commute with z and fix each cycle of z. Hence, each element in the kernel is a element of order p, and the kernel is a p-group.

Theorem

Let $z \in C$ be central and of order p. Let π be the partition of Ω given by the cycles of z, and let $C|_\pi$ be the homomorph of C that acts on the cycles. Let

$$f : C \rightarrow C|_\pi$$

be the blocks homomorphism that maps g to its action on the cycles. If $S \in Syl_p(C|_\pi)$ then $f^{-1}(S) \in Syl_p(C)$.

A straightforward recursive application of this result leads to Algorithm 1.

Algorithm 1 : Sylow Using Centralizers

```
function sylow( G:group; p:prime ):group;
(* Return a Sylow p-subgroup of the permutation group G *)
begin
  if p does not divide the order of G then
    result is <identity>;
  end if;
  find an element z of order p where C = C_G(z) contains a Sylow p-subgroup of G;
  π := cycles of z;
  let f : C → C|_π be the blocks homomorphism;
  result is f^-1( sylow( C|_π, p ) );
end;
```

We define the Sylow p-subgroup to be the identity subgroup when p does not divide the order of G in order to simplify the statement of Algorithm 1. In such cases, the result S is the kernel of the homomorphism f. There are additional termination criteria for the recursion that help improve efficiency. Suppose the order of the required Sylow subgroup is p^m. If $m = 1$, then the subgroup $<z>$ is a Sylow p-subgroup. If C is a p-group, then C is a Sylow p-subgroup.

As an example, consider the fourth group G of Chapter 10. The group G has degree 21 and is generated by

a=(1,8,9)(2,11,15)(3,10,12)(4,14,19)(5,16,17)(6,21,20)(7,13,18),
b=(9,18,20)(12,19,17), and
c=(10,21,11)(13,16,14).

It has order $27{,}783 = 3^4 \times 7^3$. A base for the group is

[1, 9, 8, 10, 2, 12]

and a strong generating set relative to this base is

$s_1 = a$, $s_2 = b$, $s_3 = c$,
s_4=(8,13,21)(10,14,16), s_5=(2,6,3)(4,5,7), and s_6=(12,20,15)(17,19,18).

To the compute the Sylow 7-subgroup, we could choose the element z = (9,15,12,19,20,17,18), which has a centralizer C of order $3087 = 3^2 \times 7^3$ which is generated

by

(10,21,11)(13,16,14), (8,13,21)(10,14,16),
(2,3,6)(4,7,5), (9,18,17,20,19,12,15), and (1,7,6)(3,4,5).

The partition π is

{ 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9,12,15,17,18,19,20 | 10 | 11 | 13 | 14 | 16 | 21 }

Let f be the corresponding homomorphism. The kernel of f is just $<z>$ of order 7. The image is a group I of degree 15 and order $441 = 3^2 \times 7^2$. I is generated by

(10,15,11)(12,14,13), (2,3,6)(4,7,5),
(1,7,6)(3,4,5), and (8,12,15)(10,13,14).

Applying the algorithm recursively to I, we find a p-central element z_I = (1,2,3,4,6,5,7)(8,11,10,13,15,14,12), whose centralizer in I is a Sylow 7-subgroup of order 7^2. It is generated by

(8,12,14,15,13,10,11), and (1,7,5,6,4,3,2).

The preimage is a Sylow 7-subgroup of G of order 7^3 and it is generated by

(8,11,10,14,21,16,13), (1,2,3,4,6,5,7), and (9,15,12,19,20,17,18).

Restricting to Orbits

Rather than use a single homomorphism and require that the kernel is a p-group, we can use several homomorphisms so that their combined use will remove any non-p part of the final preimage. What we require is two or more homomorphisms such that the intersection of their kernels is the identity. Then successive application of them will eventually result in a trivial kernel.

Theorem

Let $\Omega = \Gamma_1 \cup \Gamma_2$ where Γ_1 and Γ_2 are invariant under the action of G. Let $G|_{\Gamma_1}$ be the restriction of G to the set Γ_1. Let f_1 be the natural homomorphism

$$f_1 : G \rightarrow G|_{\Gamma_1}$$

Let $S_1 \in Syl_p(G|_{\Gamma_1})$, let S be $f_1^{-1}(S_1)$, and let f_2 be the natural homomorphism

$$f_2 : S \rightarrow S|_{\Gamma_2}$$

If $S_2 \in Syl_p(f_2(S))$ then $f_2^{-1}(S_2) \in Syl_p(G)$.

Algorithm 2 : Sylow Using Centralizers and Orbits

```
function sylow( G:group; p:prime ):group;

(* Return a Sylow p-subgroup of the permutation group G *)

begin
  if p does not divide the order of G then
    result is <identity>;
  end if;

  find an element z of order p where C = C_G(z) contains a Sylow p-subgroup of G;

  if z is fixed point free then
    result is central_sylow( C, z, p );
  else
    Γ2 := fixed points of z;
    Γ1 := Ω - Γ2;
    let f1 : C → C |Γ1 be natural homomorphism;
    S := f1^-1( central_sylow( f1(C), f1(z), p ) );
    let f2 : S → S |Γ2 be natural homomorphism;
    result is f2^-1( sylow( f2(S), p ) );
  end if;
end;

function central_sylow( G:group; z:element; p:prime ):group;

(* Given an element z of order p that is fixed point free and
 central in G, return a Sylow p-subgroup of G *)

begin
  if degree of G is p then
    result is <z>;
  else if transitive(G) then
    π := partition of Ω determined by the cycles of z;
    let f : G → G |π be the blocks homomorphism;
    result is f^-1( sylow( G |π, p ) );
  else
    Γ1 := nontrivial orbit of G;
    Γ2 := Ω - Γ1;
    let f1 : G → G |Γ1 be natural homomorphism;
    S := f1^-1( central_sylow( f1(G), f1(z), p ) );
    let f2 : S → S |Γ2 be natural homomorphism;
    result is f2^-1( central_sylow( f2(S), f2(z), p ) );
  end if;
end;
```

The function *central_sylow* requires the element z to be fixed-point-free so that its restrictions $f_1(z)$ and $f_2(z)$ are assured of not being the identity element. Hence, the restrictions are p-central in the quotient, and we can directly call *central_sylow* rather than *sylow*.

The recursion in function *central_sylow* means that we restrict to each nontrivial orbit of G in the case where G is intransitive. This could be done iteratively.

There are some ways to improve efficiency. If the preimage S under f_1 is a p-group - that is, the kernel of f_1 just happens to be a p-group - then S is a Sylow p-subgroup and it is not necessary to form f_2 and compute the Sylow subgroup of its image.

Let us again consider the Sylow 7-subgroup of the example group G. Again we choose the element z = (9,15,12,19,20,17,18), which has centralizer C of order $3087 = 3^2 \times 7^3$ which is generated by

(10,21,11)(13,16,14), (8,13,21)(10,14,16),
(2,3,6)(4,7,5), (9,18,17,20,19,12,15), and (1,7,6)(3,4,5).

The fixed points of z are

Γ_2 = {1,2,3,4,5,6,7,8,10,11,13,14,16,21},

and the non-fixed points are

Γ_1 = {9,15,12,19,20,17,18}.

The homomorphism f_1 has a kernel of order $441 = 3^2 \times 7^2$ generated by

(10,21,11)(13,16,14), (8,13,21)(10,14,16),
(2,3,6)(4,7,5), and (1,7,6)(3,4,5).

while the image is a cyclic group of order 7 generated by the image of z. Hence, S is the centralizer C.

The homomorphism f_2 of S has a kernel of order 7 generated by z. The image is a group of degree 14 and order $441 = 3^2 \times 7^2$. The image I is generated by

(9,14,10)(11,13,12), (8,11,14)(9,12,13),
(2,3,6)(4,7,5), and (1,7,6)(3,4,5).

Applying the algorithm recursively to I, we find a p-central element z_I = (1,2,3,4,6,5,7)(8,11,13,14,12,9,10), whose centralizer in I is a Sylow 7-subgroup of order 7^2. It is generated by

(8,10,9,12,14,13,11), and (1,7,5,6,4,3,2).

The preimage is a Sylow 7-subgroup of G of order 7^3 and generated by

(8,11,10,14,21,16,13), (1,2,3,4,6,5,7), and (9,15,12,19,20,17,18).

Finding Centralizers

So far we have ignored the problem of locating a p-central element in G. In practice, an element of order p is sought by examining the orders of the terms in a random sequence of elements until one is found having order divisible by p. By taking a suitable power of such an element, one obtains an element x of order p. By computing the centralizer $C_G(x)$, we can determine whether or not x is p-central.

In the case of simple and near-simple groups G, such a random search will often locate a p-central element after generating two or three random elements. However, for other classes of groups, such as soluble groups, such a random method may have to examine an impractically large number of elements in order to have any chance of locating a p-central element.

In such cases we may resort to the following algorithm:

Algorithm 3 : Locating p-Central Elements

```
function p_central( G:group; p:prime ):element;
(* Given a group G of order p^m s, where m>0 and p does not divide s,
return a p-central element of G *)
begin
    x := a randomly chosen element of order p;
    C := C_G(x);
    let p^r be the largest power of p dividing |C|;
    while r ≠ m do
        P := Sylow p-subgroup of C;
        x := an element of Z(P) of order p such that p^(r+1) divides |C_G(x)|;
        C := C_G(x);
        let p^r be the largest power of p dividing |C|;
    end while;
    result is x;
end;
```

Consider the group G of order $27{,}783 = 3^4 \times 7^3$ for the prime p=3. There are 4256 elements of order p, of which only 686 are p-central. There are 12,348 elements which power to a p-central element, and 10,837 elements which power to an element of order 3 which is not p-central. There are only 343 elements whose order is not divisible by 3. Hence, the chances that a random element will power to a p-central element is approximately 50% and all is well for this group.

If, however, we did follow Algorithm 3 and chose a non-p-central element x of order 3, then the centralizer C has a Sylow 3-subgroup P of order 3^3 which is elementary abelian. Hence, the centre of P is the whole of P. Of the 26 elements in $Z(P)$ only 2 are p-central, so the algorithm may need to consider 25 elements before locating a p-central one.

Random Elements

In determining random elements of the group, we wish to obtain each element with equal probability. For a group G with a base and strong generating set this can be done by independently choosing a random coset representative $u_i \in U^{(i)}$, for each i, and taking the random element $u_k \times u_{k-1} \times \cdots \times u_1$. Selecting a random coset representative requires randomly choosing an integer in the range 1 to $|U^{(i)}|$ with uniform distribution, or equivalently choosing a random point in the set $\Delta^{(i)}$.

For a group G given only by a set of generators S the situation is difficult, and we cannot guarantee a uniform distribution of random elements. One approach used is to consider a random word w in the generators S of a given length and then evaluating the word. Each symbol in w is chosen randomly from the set S (and perhaps the set of inverses). It is a simple enough task to choose each symbol of w randomly with uniform distribution. However, there are two major problems:

1. To ensure that it is possible to generate each element of the group in this way requires excessively long words: for example, the dihedral group of degree n and order $2 \times n$ is generated by two elements - $s_1 = (1,n)(2,n-1)(3,n-2)...(n/2,n/2+1)$ and $s_2 = (1,n-1)(2,n-2)(3,n-3)...(n/2-1,n/2+1)$ - and the element $(s_1 \times s_2)^{n/2}$ can not be generated by a word of length less than n.
2. Even if we can generate each element of G by words of the specified length there is no guarantee that the distribution is uniform.

The long words required are also expensive to evaluate.

An approach used to reduce this expense is to only use long words as seeds to the random generator. Each seed w is evaluated to an element g and used to generate a small number of elements by randomly choosing a generator s by which to multiply the current element. After this short sequence of random elements has been generated, a new seed is chosen.

Algorithm 4 : Random Elements

Input: a group G given by a set of generators S;

Output: a sequence of pseudo-random elements of G;

```
begin
    determine l, the length of words required, from |S| and |Ω|;
    while true do
        choose a random word w of length l;
        evaluate w to obtain g;
        for i := 1 to small_number do
            choose a random generator s ∈ S;
            g := g × s;  (* the next random element *)
        end for;
    end while;
end;
```

More on Blocks Homomorphisms

We have previously used a block homomorphism which was guaranteed to have a p-group as a kernel. However, we can follow the lead of the transitive constituent homomorphisms and choose two homomorphisms whose kernels have trivial intersection. If two block systems are distinct and minimal then the corresponding blocks homomorphisms will have trivial intersection, and we can utilise the following result.

Theorem

Let G be transitive on Ω and let ρ_1 and ρ_2 be distinct minimal systems of imprimitivity of G. Let $G\,|_{\rho_1}$ be the action of G induced on the subsets of ρ_1. Let f_1 be the natural homomorphism

$$f_1 : G \rightarrow G\,|_{\rho_1}$$

Let $S_1 \in Syl_p(G\,|_{\rho_1})$, let S be $f_1^{-1}(S_1)$, and let f_2 be the natural homomorphism

$$f_2 : S \rightarrow S\,|_{\rho_2}$$

If $S_2 \in Syl_p(f_2(S))$ then $f_2^{-1}(S_2) \in Syl_p(G)$.

There is some possibility to reduce the problem in the case where G has only one minimal block system. It just may be the case that the kernel is a p-group, or the preimage of a Sylow p-subgroup of its image may be a proper subgroup of G. Provided that G itself is not a p-group, these techniques will fail to reduce the problem precisely when the image is a p-group. Other techniques (like those of Algorithm 1) must then be used.

If the group is primitive then there are no block systems to use. If the degree of G is not divisible by p, then the point stabiliser contains a Sylow p-subgroup, and we can reduce the problem.

Algorithm 5 : Sylow Using Orbits and Blocks

```
function sylow( G:group; p:prime ):group;

(* Return a Sylow p-subgroup of the permutation group G *)

begin
  if p does not divide the order of G then
    result is <identity>;
  end if;

  if not transitive(G) then
    for each orbit Γ of G do
      let f : S → S |Γ be natural homomorphism;
      S := f^-1( sylow( f(S), p ) );
      if S is a p-group then break; end if;
    end for;
    result is S;
  else
    result is transitive_sylow( S, p );
  end if;
end;

function transitive_sylow( G:group; p:prime ):group;

(* Return a Sylow p-subgroup of a transitive permutation group G *)

begin
  if degree of G is not divisible by p then
    result is sylow( G1, p );
  else if G has minimal block systems ρ1, ρ2 then
    let f1 : G → G |ρ1 be natural homomorphism;
    S := f1^-1( sylow( f1(G), p ) );
    let f2 : S → S |ρ2 be natural homomorphism;
    result is f2^-1( sylow( f2(S), p ) );
  else if G has minimal block system ρ1 then
    let f1 : G → G |ρ1 be natural homomorphism;
    S := f1^-1( sylow( f1(G), p ) );
    if S = G then
      (* algorithm fails *)
    else
      result is sylow( S, p );
    end if;
  else (* G is primitive and p divides degree *)
    (* algorithm fails *)
  end if;
end;
```

As an example, again consider the fourth group G of Chapter 10. The group G has degree 21 and is generated by

a=(1,8,9)(2,11,15)(3,10,12)(4,14,19)(5,16,17)(6,21,20)(7,13,18),
b=(9,18,20)(12,19,17), and
c=(10,21,11)(13,16,14).

It has order $27{,}783=3^4 \times 7^3$. The group is imprimitive with one minimal block system ρ given by

{ 1,2,3,4,5,6,7 | 8,10,11,13,14,16,21 | 9,12,15,17,18,19,20 }.

Let f be the corresponding homomorphism. The kernel of f is a subgroup K of order $9261 = 3^3 \times 7^3$ generated by

(9,18,20)(12,19,17), (10,21,11)(13,16,14),
(12,20,15)(17,19,18), (1,7,5,6,4,3,2),
(8,13,21)(10,14,16), (1,7,3)(2,4,6), and (2,3,6)(4,7,5).

The image is a cyclic group of order 3, so by convention it has trivial Sylow 7-subgroup, and the preimage S is just K.

The algorithm is applied recursively to K. The group is intransitive with three orbits of length 7 corresponding to the three blocks of ρ. Let f_1 be the homomorphism of K corresponding to the orbit {1,2,3,4,5,6,7}. The kernel of f_1 has order $441 = 3^2 \times 7^2$ and is generated by

(9,18,20)(12,19,17), (10,21,11)(13,16,14),
(12,20,15)(17,19,18), and (8,13,21)(10,14,16).

The image is a group of order $21 = 3 \times 7$ which has a Sylow 7-subgroup generated by (1,2,3,4,6,5,7). The preimage of this Sylow 7-subgroup is S of order $3^2 \times 7^3$ generated by

(1,2,3,4,6,5,7), (9,18,20)(12,19,17), (8,13,21)(10,14,16),
(10,11,21)(13,14,16), and (12,15,20)(17,18,19).

Let f_2 be the homomorphism of S corresponding to the orbit {8,10,11,13,14,16,21}. The kernel of f_2 has order $147 = 3 \times 7^2$ and is generated by

(1,2,3,4,6,5,7),
(9,18,20)(12,19,17), and (12,15,20)(17,18,19).

The image is a group of order $21 = 3 \times 7$ which has a Sylow 7-subgroup generated by (1,2,7,4,3,5,6). The preimage of this Sylow 7-subgroup is S of order 3×7^3 generated by

(8,14,13,10,16,11,21), (1,2,3,4,6,5,7),
(9,18,20)(12,19,17), and (12,15,20)(17,18,19).

Let f_3 be the homomorphism of S corresponding to the orbit {9,12,15,17,18,19,20}. The kernel of f_3 has order $49 = 7^2$ and is generated by

(8,14,13,10,16,11,21), and (1,2,3,4,6,5,7).

The image is a group of order $21 = 3 \times 7$ which has a Sylow 7-subgroup generated by (1,2,7,5,3,6,4). The preimage of this Sylow 7-subgroup is S of order 7^3 generated by

(9,19,18,12,17,15,20), (8,14,13,10,16,11,21), and (1,2,3,4,6,5,7).

Primitive Groups

All is not lost if the group G is primitive. The primitive permutation groups have been classified by what is called the O'Nan-Scott Theorem into very well-specified cases. However, to date only some of the cases have practical solutions to reducing the problem of computing Sylow p-subgroups, and we will not pursue them in this chapter.

Summary

This chapter has demonstrated how homomorphisms can be used to reduce the problem of computing a Sylow p-subgroup to smaller cases. This divide-and-conquer approach to analysing permutation groups and solving problems is very general and powerful. We will see further examples in later chapters.

The reductions mentioned in this chapter can be combined with each other and combined with the cyclic extension method or methods for treating soluble permutation groups. The best mix is a matter of engineering and is still the focus of further investigation.

Exercises

(1/Easy) Exercise 1(i) of Chapter 11 asks you to compute the centralizer of z=(1,2)(4,7) in the symmetries G of the projective plane of order 2. This element is p-central, for $p = 2$. Use the centralizer to compute a Sylow 2-subgroup of G.

(2/Easy) Exercise 1(ii) of Chapter 11 asks you to compute the normalizer of < (1,2)(4,7), (4,7)(5,6) > in the symmetries G of the projective plane of order 2. Restrict the normalizer to its action on the orbit of length 3 and hence compute a Sylow 2-subgroup of G.

(3/Easy) Find a 2-central element of the symmetric group of degree 4, compute its centralizer, and compute a Sylow 2-subgroup.

(4/Moderate) Let G be the symmetries of the projective plane of order 2. Execute Algorithm 5 for this example and the prime $p = 2$. As p does not divide the degree 7, we work within G_1 and restrict G_1 to its orbit of length 6. The image has one block system and the kernel of the corresponding homomorphism is a group of order 4. Take a random element of order 2 in the image of the blocks homomorphism and obtain a Sylow 2-subgroup of this image. Hence, compute a Sylow 2-subgroup of G.

(5/Moderate) Exercise 3 of Chapter 15 deals with a group G of degree 14 and order $2^9 \times 3 \times 7$, which is the sixth group of Chapter 10. G has one block system and the corresponding homomorphism has a kernel of order 2^6. Use the information in Exercise 3 of Chapter 15 and compute a Sylow 2-subgroup of G.

Bibliographical Remarks

The problem of constructing a Sylow subgroup for a moderately large group was first considered in J.J. Cannon, *"Computing local structure of large finite groups"*, **Computers in Algebra and Number Theory,** G. Birkhoff and M. Hall, Jr (eds), (SIAM-AMS Proceedings, Vol. 4), American Mathematical Society, Providence, R.I., 1971, pp. 161-176. The Sylow algorithm presented in that paper works for general groups. It is based on the cyclic extension technique (of chapter 6). It begins by finding a p-element, and searches centralizers for the extending elements. The algorithm can find Sylow subgroups in groups of order up to one million.

This algorithm was implemented for permutation groups by the author in 1977 to specifically use the stabiliser chain of the centralizer during a backtrack search for extending elements and to find a strong generating set. The work is described in the author's Ph.D. thesis of 1979 and in G. Butler and J.J. Cannon, *"Computing in permutation and matrix groups III: Sylow subgroups"*, Journal of Symbolic Computation, **8** (1989) 241-252. This backtrack method for computing Sylow subgroups begins to struggle on permutation groups of degrees in the hundreds, or those groups whose order is divisible by a high power of p, say p^{10} or higher. The techniques described in this chapter using homomorphisms were introduced to overcome these limitations. The use of homomorphic images of centralizers is first studied in G. Butler and J.J. Cannon, *"Using homomorphisms to compute Sylow subgroups of permutation groups"*, TR 222, Basser Department of Computer Science, University of Sydney, 1984, along with limited use of restricting to orbits in order to reduce the degree of the group. The bottleneck of finding a p-element z was particularly evident in soluble groups, so further development of this approach had to wait for the work of S.P. Glasby, *"Computing normalisers in finite soluble groups"*, Journal of Symbolic Computation, **5** (1988) 285-294, on computing a Sylow subgroup of a soluble group given by an AG-system and the work (in 1987 and 1989 respectively) of C.C. Sims, *"Computing the order of a solvable permutation group"*, Journal of Symbolic Computation, **9** (1990) 699-705, and G. Butler, *"Computing a conditioned pc presentation of a soluble permutation group"*, TR 392, Basser Department of Computer Science, University of Sydney, 1990, to set up the isomorphism between a soluble permutation group and an appropriate AG-system (see chapter 18 for more details). The resulting algorithm is presented in G. Butler and J.J. Cannon, *"Computing Sylow subgroups of permutation groups via homomorphic images of centralizers"*, to appear in Journal of Symbolic Computation.

Around 1982, Derek Holt at Warwick implemented an algorithm which searches the normalizer of a p-group for the extending elements required by the cyclic extension method. It works up the stabiliser chain of the group G and begins with the first level of index divisible by p. A heuristic looks at random elements from the current level of the chain for one that can be powered to a p-element which extends the p-group. If the heuristic fails then the normalizer is computed and searched for one extending element. The process is repeated until a Sylow subgroup of the stabiliser is found, and then the algorithm proceeds to the next higher level whose index is divisible by p. This work is hinted at in D.F. Holt, *"A computer program for the calculation of the Schur multiplier of a permutation group"*, **Computational Group Theory,** M.D. Atkinson (ed.), Academic Press, Academic Press, 1984, pp. 307-319, and D.F. Holt, *"Computing normalizers in permutation groups"*, to appear in Journal of Symbolic Computation.

More use of the natural homomorphisms of permutation groups was made by M.D. Atkinson and P.M. Neumann, *"Computing Sylow subgroups of permutation groups"*, (Twenty-first Southeastern Conference on Combinatorics, Graph Theory, and Computing, Boca Raton, February 1990), to appear in Congressus Numeratium. In particular, they exploit the fact a group may have two minimal block systems, and the fact that a primitive group may have an elementary abelian regular normal p-subgroup. In several cases, they restrict to the point stabiliser. In one case, the Sylow subgroup of the point stabiliser is used as the starting point in the cyclic extension method. In that case, they extend the p-group using the techniques of Holt described above. However, there are cases where their approach cannot proceed and must fall back on an alternative way of computing Sylow subgroups.

The O'Nan-Scott Theorem and its use in computation with permutation groups is explained in P.M. Neumann, *"Some algorithms for computing with permutation groups"*, **Groups - St Andrews 1985**, E.F. Robertson and C.M. Campbell (eds), London Mathematics Society Lecture Notes **121**, Cambridge University Press, Cambridge, 1986, pp. 59-92.

There have also been several theoretical investigations of the complexity of the problem. These rely on the classification of simple groups, reductions using homomorphisms, and the recognition of simple groups to give polynomial-time algorithms for computing Sylow subgroups of permutation groups. In the general case, the complexity is $O(|\Omega|^9)$. W.M. Kantor, *"Polynomial-time algorithms for finding elements of prime order and Sylow subgroups"*, Journal of Algorithms, **6** (1985) 478-514, restricts to the case where the group G is simple, while W.M. Kantor and D.E. Taylor, *"Polynomial-time versions of Sylow's theorem"*, Journal of Algorithms, **9** (1988) 1-17, restrict to a group G which is either soluble or has all of its noncyclic composition factors suitably restricted. The general case is treated in W.M. Kantor, *"Sylow's theorem in polynomial-time"*, Journal of Computer Systems and Science, **30** (1985) 359-394. Glasby's algorithm for computing the Sylow subgroup of a soluble group defined by an AG-system is based on Kantor's ideas.

Knowledge of the Sylow p-subgroup of G for various primes p dividing the order of G is crucial when analyzing the structure of G. In particular, the construction of a Sylow p-subgroup is a basic step in current algorithms for computing such things as the maximal normal p-subgroup of G, $O_p(G)$; the Fitting subgroup; the socle of G; representatives of conjugacy classes of elements of prime power order; and the first and second cohomology groups of G.