Performance of the GAP-function Normalizer and an attempt of its improvement II

Izumi Miyamoto
University of Yamanashi

Let $\Omega = \{1, 2, \cdots, n\}$.

Let G and H be permutation groups on Ω.

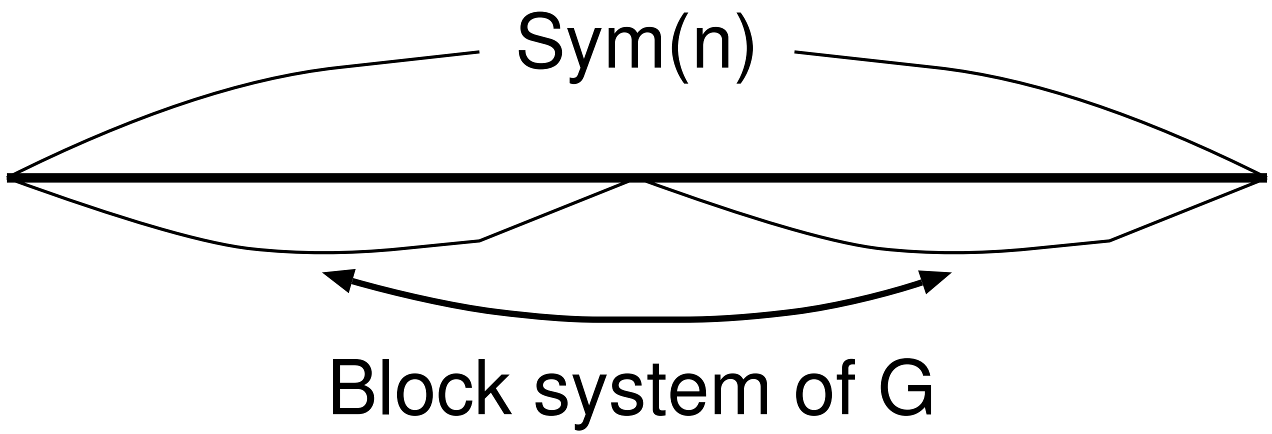
The normalizer of G in H is defined by

$$\text{Norm}(H, G) = \{\ h \in H | h^{-1} G h = G\ \}.$$

Suppose $H = Sym(n)$ =SymmetricGroup(Ω).

GAP4 - Groups, Algorithms, Programming (version 4)- a System for Computational Discrete Algebra has a special function "DoNormalizerSA" for such cases.

If G is imprimitive and only one block system of block length l, for instance, $l = 2$,



then
$$\text{Norm}(Sym(n), G) \subseteq W,$$

where $W =$WreathProduct$(Sym(n/2), Sym(2))$.

DoNormalizerSA invokes NormalizerParentSA to compute W and then computes

$$\text{Norm}(W, G) \;\text{ instead of Norm}(Sym(n), G).$$

Even if G is primitive, we have such a subgroup as above.

<u>*Proposition.*('97)</u>

If G is transitive, then the normalizer of G is contained in the automorphism group of the <u>*association scheme*</u> A formed by G.

So
$$\mathrm{Norm}(Sym(n), G) = \mathrm{Norm}(Aut(A), G).$$

The wreath product W appears as the automorphism group of a typical association scheme.

Example: the relation matrix of an association scheme consisting of the orbits of G on $\Omega \times \Omega$

$$A = \left.\overbrace{\begin{pmatrix} 0&1&1&1&1&1&2&2&2&2&2&2\\ 1&0&1&1&1&1&2&2&2&2&2&2\\ 1&1&0&1&1&1&2&2&2&2&2&2\\ 1&1&1&0&1&1&2&2&2&2&2&2\\ 1&1&1&1&0&1&2&2&2&2&2&2\\ 1&1&1&1&1&0&2&2&2&2&2&2\\ 2&2&2&2&2&2&0&1&1&1&1&1\\ 2&2&2&2&2&2&1&0&1&1&1&1\\ 2&2&2&2&2&2&1&1&0&1&1&1\\ 2&2&2&2&2&2&1&1&1&0&1&1\\ 2&2&2&2&2&2&1&1&1&1&0&1\\ 2&2&2&2&2&2&1&1&1&1&1&0 \end{pmatrix}}^{\Omega}\right\}\Omega$$

$$Aut(A) = \text{WreathProduct}(Sym(6), Sym(2))$$

Example: the relation matrices of association schemes

$$A = \begin{pmatrix} 0 & 1 & 1 & 2 & 2 & 2 \\ 1 & 0 & 1 & 2 & 2 & 2 \\ 1 & 1 & 0 & 2 & 2 & 2 \\ 2 & 2 & 2 & 0 & 1 & 1 \\ 2 & 2 & 2 & 1 & 0 & 1 \\ 2 & 2 & 2 & 1 & 1 & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 0 & 1 & 2 & 3 & 3 & 3 \\ 2 & 0 & 1 & 3 & 3 & 3 \\ 1 & 2 & 0 & 3 & 3 & 3 \\ 3 & 3 & 3 & 0 & 1 & 2 \\ 3 & 3 & 3 & 2 & 0 & 1 \\ 3 & 3 & 3 & 1 & 2 & 0 \end{pmatrix}$$

WreathProduct($Sym(3), Sym(2)$) forms A.

WreathProduct($Cyc(3), Sym(2)$) forms B.

Both groups have only one same block system.

Block system cannot distinguish A and B.

We would like to show two algorithms

Algorithm A-I and Algorithm A-II,

which only work on transitive groups now.

We will not use association schemes but only Wreath-Products in both Algorithms. Our programs are short and consist of 100 lines or so.

We use a backtrack method to compute the automorphism groups of association schemes. So it is not an easy computation, but it is much easier than to compute normalizers directly in some cases.

If G is transitive, we can also use the following lemma.

<u>*Lemma.*('00)</u>

Let K be a permutation group on Ω. Let F be a tuple $[p_1, p_2, \cdots, p_r]$ of points in Ω and let G^i be the stabilizer of the subset $[p_1, p_2, \cdots, p_i]$ of F as a tuple in G for $i = 1, 2, \cdots, r$. Let I^i be the group of isomorphisms of the system of association schemes of G^i on $\Omega \backslash [p_1, p_2, \cdots, p_i]$. Set $I^0 = I$, $G^0 = G$ and set $I^{\{0..i\}} = I^0 \cap I^1 \cap \cdots \cap I^i$. Suppose that $G^i \cap K$ is transitive on the orbit of $I^{\{0..i\}} \cap K$ containing the point p_{i+1} for $i = 0, 1, \cdots, r-1$. Then the normalizer of G in K is generated by $G \cap K$ and the normalizer of G in $I^{\{0..r\}} \cap K$.

K in Lemma is used instead of $Sym(n)$ in Norm$(Sym(n), G)$.

K may be WreathProduct or $Aut(A)$.

Lemma says

if K and G have a same orbit, containing p,

then Norm$(K, G) = G$Norm(K_p, G).

Note that

Norm(K_p, G) =Norm(Norm$(K_p, G_p), G)$.

Suppose furthermore that

if K_p and G_p have a same orbit, containing p',

then Norm$(K_p, G_p) = G_p$Norm$(K_{p,p'}, G_p)$.

Norm$(K_{p,p'}, G_p)$ =Norm(Norm$(K_{p,p'}, G_{p.p'}), G_p)$.

Norm$(K, G) = G$Norm(Norm$(K_{p,p'}, G_p), G)$.

Or something else happens so that

OrbitLengths$(G_p) = [l_1, l_2]$ $(l_1 \neq l_2)$.

Then Norm(K_p, G_p) =Norm$(K_p \cap D, G_p)$,

where D =DirectProduct$(Sym(l_1), Sym(l_2))$.

(D is computed by NormalizerParentSA.)

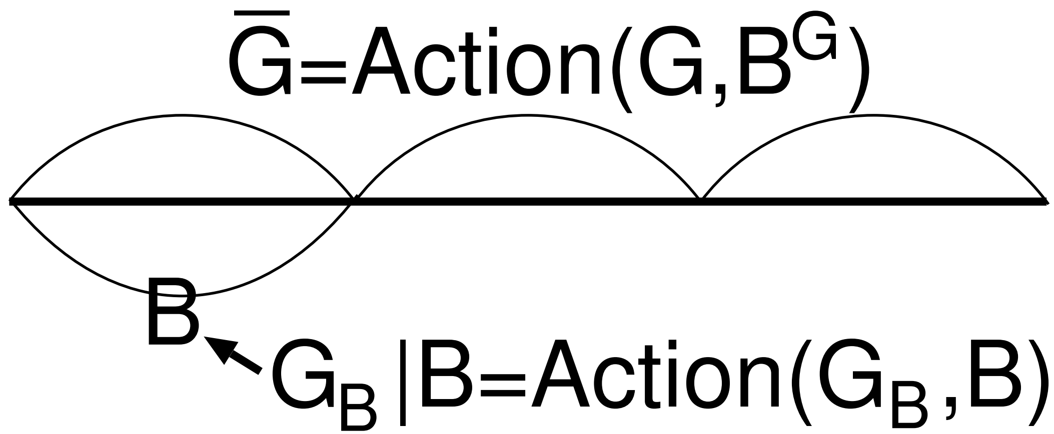
So various heuristics arise from this computation.

From these arguments we have

Algorithm A-I which uses a lot of heuristics.

Suppose that

B^G is the only one block system of G of $l = |B|$.



Here G_B is the setwise stabilizer of B in G.

SubgpConjSymmgp, another GAP special function, which computes a conjugating element between two subgroups in a symmetric group considers these actions $\bar{G}$ and $G_B|B$.

For computing normalizers, let

$H =$Norm$(Sym(l), G_B|B)$ and $K =$Norm$(Sym(n/l), \bar{G})$.

Then

$$\text{Norm}(Sym(n), G) \subseteq \text{WreathProduct}(H, K)$$

We apply this argument recursively. We use Lemma on this WreathProduct(H, K) with some heuristics.

This gives <u>Algorithm A-II</u>

Experiment

$$\text{Norm}(Sym(n), \text{TransitiveGroup}(n, k))$$

of degree n, $20 \leq n \leq 30$.

	number of groups
$20 \leq n \leq 30$	36,620
WreathProduct	36,413
primitive	105
remaining	102

Computing times of the normalizers of transitive groups of degree n, $20 \leq n \leq 30$, in $Sym(n)$

time range	DoNorm	A-I	A-II
$* \leq$0.2sec	22238	956	85
0.2sec $< * \leq$0.5sec	5433	24213	1575
0.5sec $< * \leq$1sec	2200	10377	15698
1sec $< * \leq$3sec	1572	788	18994
3sec $< * \leq$10sec	1162	170	237
10sec $< * \leq$40sec	1005	39	31
40sec $< * \leq$5min	1176	66	0
5min $< * \leq$30min	705	9	0
30min $< * \leq$1h	114	2	0
1h $< * \leq$10h	260	0	0
10h $< *$?	755	0	0
total time	?	10.6h	11.4h

DoNormalizerSA and AutomorphismGroupPermGroup

which computes Norm($Sym(n), G$) directly. We computed 31091 Norm($Sym(n), G$)'s within 2 hours each.

(in minutes)

time range by AutPerm	number	total time by DoNorm	total time by AutPerm
$*\ \geq$0sec	31091	18428	16599
$*\ \geq$1sec	3973	18334	16543
$*\ \geq$10sec	2180	18189	16439
$*\ \geq$1min	1341	17626	16047
$*\ \geq$10min	391	13548	12869
$*\ \geq$60min	62	4988	4817
$*\ \geq$90min	11	1165	1139

DoNormalizerSA and AutomorphismGroupPermGroup

applied to

some intransitive groups of degree $n-1$

$\mathrm{Norm}(Sym(n-1), \mathrm{Stabilizer}(\mathrm{PrimitiveGroup}(n,k), n))$

(in seconds)

n	k	DoNorm	AutPerm
81	123	37	0.2
100	3	77	0.3
105	9	12	0.3
112	1	19652	0.4
120	12	46	0.5

G =Stabilizer(PrimitiveGroup(81,123),81)

OrbitLengths(G) $= [40, 40]$

W =WreathProduct(Sym(40)$,$ Sym(2))

It took 23 seconds for SmallGeneratingSet(W).

G =Stabilizer(PrimitiveGroup(112,1),112)

OrbitLengths(G) $= [81, 30]$

W =DirectProduct(Sym(81),DoNorm(Sym(30)$, G^{O_2}$))

$G^{O_2} \cong$TransitiveGroup(30, 1019)

It took 11854 seconds for DoNorm(Sym(30)$, G^{O_2}$).

It took 0.2 seconds for DoNorm(Sym(81)$, G^{O_1}$).

G is faithful on both orbits.

Example: Norm(H, G), $H \neq Sym(n)$

H =WreathProduct$\not\supseteq$ G =TransitiveGroup(n, k)

time range by DoNorm$(Sym(n), G)$	30min$\leq * \leq$1hour
number of groups	117
total time for DoNorm$(Sym(n), G)$	4962min
total time for Norm(H, G)	1162min

Some computing times for DoNorm$(Sym(n), G)$ and

Norm(H, G) (in seconds)

n	k	DoNorm$(Sym(n), G)$	Norm(H, G)
30	2173	3565	6052
30	2256	2064	2580
30	2548	2714	3200
30	2560	2704	3311
30	4644	1978	4230

Remark : Computational complexity of Normalizer

L.M. Luks and T. Miyazaki
Polynomial-time normalizers for permutation groups with restricted composition factors. (ISSAC2002)

$$\mathrm{Norm}(H, G)$$

If H has restricted composition factors, then Norm is $\mathcal{P}$.
This algorithm seems far from actual computation now.

References:

I. Miyamoto. Performance of the GAP-function Normalizer and an attempt of its improvement ftp://tnt.math.met
u.ac.jp/pub/ac05/miyamoto/