Efficient characteristic refinements for finite groups

Joshua Maglione

Department of Mathematics, Colorado State University, Fort Collins, CO 80523, USA

Abstract

Filters were introduced by J.B. Wilson in 2013 to generalize work of Lazard with associated graded Lie rings. It holds promise in improving isomorphism tests, but the formulas introduced then were impractical for computation. Here, we provide an efficient algorithm for these formulas, and we demonstrate their usefulness on several examples of p-groups.

Key words: p-groups, isomorphism, Lie algebras, filters

1. Introduction

Isomorphism between two finite groups becomes easier when we use isomorphism invariant subgroups (i.e. characteristic subgroups) to constrain the number of possibilities. With this in mind, Fitting uncovered several characterististic subgroups to later be used to determine isomorphism between groups (Fitting, 1938), see the accompanying bibliography in (Cannon and Holt, 2003). However, in the case of p-groups, these characteristic subgroups are usually the whole group or the trivial group. As seen in (Eick et al., 2002), the inclusion of just one new characteristic subgroup can greatly improve performance.

New sources for computable characteristic subgroups of p-groups were uncovered in (Wilson, 2013, 2015). In addition, it was shown that the inclusion of new characteristic subgroups induced more subgroups and gave formulas to automate this process of refining. However, the formulas required an exponential amount of computation. In this paper, we prove that we can do this in polynomial time and we provide an implementation for Magma. Indeed, even for groups of order 3^{100}, we are able to refine a typical characteristic series by about ten-fold in just a few minutes; see Figure 1 on page 12.

A *filter* for a group G is a function $\phi : M \to 2^G$ from a commutative monoid $M = \langle M, +, 0, \preceq\rangle$ into the normal subgroups of G satisfying the following: for all $s, t \in M$

$$[\phi_s, \phi_t] \leq \phi_{s+t} \qquad \& \qquad s \preceq t \Longrightarrow \phi_t \leq \phi_s.$$

Email address: `maglione@math.colostate.edu` (Joshua Maglione).

Wilson proves (Wilson, 2013, Theorem 3.1) that each filter has an associated Lie ring:

$$L(\phi) = \bigoplus_{s \in M - \{0\}} \phi_s / \langle \phi_{s+t} \mid t \in M - \{0\} \rangle . \tag{1}$$

The use of monoids M is essential as it allows for somewhat arbitrary refinements some of which are discussed in Section 4. We prove the following theorem.

Theorem 1. *Suppose $\phi : \mathbb{N}^d \to 2^G$ is a filter where $\preceq$ is the lexicographical order. If $H \lhd G$ and there exists $s \in \mathbb{N}^d$ such that*

$$\left\langle \phi_{s+t} \mid t \in \mathbb{N}^d - \{0\} \right\rangle < H < \phi_s,$$

then there exists a polynomial-time algorithm that refines ϕ to contain H in its image.

The result is smaller homogeneous components, faster automorphism computations, and an easier explanation of structure. Indeed, in (Maglione, 2015), it was shown that even well-studied unipotent classical groups admit surprises such as characteristic filters whose factors are at most of order p^2. Together with (Eick et al., 2002), this then reduces automorphism questions to $\mathrm{GL}(2, p)$ instead of $\mathrm{GL}(d, p)$. This and further uses in (Maglione, 2016; Wilson, 2015) make it desirable to compute with filters efficiently.

In addition to providing a computational framework for filters in Section 3, we refine several filters for common examples of p-groups in Section 5. We look to large examples in the literature and we also consider a sample of 2,000 sections (i.e. quotients of subgroups) of the Sylow 3-subgroups of classical groups of Lie type. We find that the larger the group, the more new structure we find, and because of the repetitive nature, often one discovery leads to more discoveries. All of our computations were run in Magma V.21-5 (Bosma et al., 1997) on a computer with Intel Xeon W3565 microprocessors at 3.20 GHz.

2. Preliminaries

We denote the set of nonnegative integers by $\mathbb{N}$, and the set of all subsets of a set G by 2^G. For groups and rings, we follow notation found in (Gorenstein, 1980). For $g, h \in G$, we set

$$[g, h] = g^{-1} g^h = g^{-1} h^{-1} g h;$$

for $X, Y \subseteq G$, we set

$$[X, Y] = \langle [x, y] : x \in X, y \in Y \rangle.$$

We let $\mathbb{Z}_p$ denote the group $\mathbb{Z}/p\mathbb{Z}$.

For a p-group G, we consider two recursively defined series: the lower central series and the exponent-p central series. The lower central series starts with $\gamma_1(G) = G$ and $\gamma_{i+1}(G) = [\gamma_i(G), G]$, and the exponent-$p$ central series begins with $\eta_1(G) = G$ and $\eta_{i+1}(G) = [\eta_i(G), G]\eta_i(G)^p$. The class ($p$-class) of G is the number of nontrivial terms in the lower central series (exponent-p central series).

2.1. Complexity

An algorithm runs in *polynomial time* if the number of operations it uses is bounded by a polynomial of the input length. At least one mark of efficiency is polynomial time, but we include run times from experiments as well.

We assume the standard models of computation for groups: permutation and matrix groups and groups given by finite presentations, see (Holt et al., 2005). All of our methods use at most a polynomial number of operations in the input size, which can be as small as $\log |G|$. They further depend on efficient methods to compute the order of subgroups and normal closures. This is in polynomial time for groups represented as permutations or matrices (Seress, 2003; Luks, 1992). For groups given by power-conjugate presentations, there are highly practical algorithms for these tasks, although some problems are not known to be in polynomial time, see (Leedham-Green and Soicher, 1990, 1998). We remark that all of our tests used power-conjugate presentations.

2.2. *Filters and prefilters*

The formulas found in (Wilson, 2013) are for commutative monoids, but for computational feasibility, we concentrate only on $\mathbb{N}^d$ with the lexicographical order. We note that every filter $\phi : \mathbb{N}^d \to 2^G$ induces a *boundary filter* $\partial\phi : \mathbb{N}^d \to 2^G$, where $\partial\phi_s = \langle \phi_{s+t} \mid 0 \prec t\rangle$. These are the factor groups of the homogeneous components of the Lie ring in equation (1).

Definition 1. A function $\pi : X \to 2^G$ is a *prefilter* if it satisfies the following conditions.
(1) There exists d such that $0 \in X \subseteq \mathbb{N}^d$ and $\langle X\rangle = \mathbb{N}^d$;
(2) if $x \in X$ and $y \in \mathbb{N}^d$ with $y \preceq x$, then $y \in X$;
(3) for all $x \in X$, $\pi_x \trianglelefteq G$;
(4) if $x, y \in X$ with $x \preceq y$ then $\pi_x \geq \pi_y$.

For $s \in \langle X\rangle$, a *partition* of s with respect to X is a sequence $(s_1, ..., s_k)$ where each $s_i \in X$ and $s = \sum_{i=1}^{k} s_i$. Let $\mathcal{P}_X(s)$ denote the set of partitions of $s \in \langle X\rangle$ with respect to X, and if $P = (s_1, ..., s_k) \in \mathcal{P}_X(s)$, then set

$$[\pi_P] = [\pi_{s_1}, ..., \pi_{s_k}].$$

For a function $\pi : X \to 2^G$, define a new function $\overline{\pi} : \langle X\rangle \to 2^G$ where

$$\overline{\pi}_s = \prod_{P\in\mathcal{P}_X(s)} [\pi_P]. \tag{2}$$

Theorem 2 (Wilson (2013, Theorem 3.3))**.** *If π is a prefilter, then $\overline{\pi}$ is a filter.*

Observe that $|\mathcal{P}_X(s)|$ is exponential in $\log |G|$, so the equation for $\overline{\pi}$ in (2) is not a practical nor a polynomial-time formula.

3. Algorithms

We have two basic problems for filters. We remind the reader that all of our filters are totally ordered, and we assume that $\mathbb{N}^d$ is ordered by the lexicographical ordering.

Problem. Evaluate
Input: A filter $\phi : \mathbb{N}^d \to 2^G$ and $s \in \mathbb{N}^d$;
Return: Generators for the subgroup ϕ_s.

Problem. Boundary

Input: A filter $\phi : \mathbb{N}^d \to 2^G$;
Return: A filter $\partial\phi : \mathbb{N}^d \to 2^G$.

After we determine polynomial-time algorithms for these problems, we discuss an algorithm for the following problem and prove it can be solved in polynomial time.

Problem. GENERATE
Input: A prefilter $\pi : \mathbb{N}^d \to 2^G$;
Return: Its closure $\overline{\pi} : \mathbb{N}^d \to 2^G$.

3.1. A data structure of filters

We introduce a data structure to compute with filters which admits polynomial-time algorithms for the three problems listed above. To do this, we assume the following properties of our filters.

Definition 2. A filter $\phi : \mathbb{N}^d \to 2^G$ is *full* if for all $1 \neq H \in \mathrm{im}(\phi)$, there exists $s \in \mathbb{N}^d$ such that $\phi_s = H$ and $\phi_s \neq \partial\phi_s$.

Said another way, we assume that $I_H = \{s \in \mathbb{N}^d : \phi_s = H\}$ has a maximal element. Since we assume a total ordering, I_H has a unique maximal element, and we denote it by m_H. Observe that a filter $\phi : \mathbb{N}^d \to 2^G$ for a finite p-group G is full if, and only if, $|\partial\phi_0| = |L(\phi)|$. This might seem like a restrictive assumption, but it is relatively harmless. We will prove that for every filter, we can construct a full filter without changing the image in Section 3.2.

The data structure of a filter $\phi : \mathbb{N}^d \to 2^G$ is a signed set that stores the pairs $(m, H) \in \mathbb{N}^d \times 2^G$ for each $1 \neq H \in \mathrm{im}(\phi)$. If for every $1 \neq H \in \mathrm{im}(\phi)$, the set I_H has a maximum, then $\mathrm{sign}(\phi) = 1$ and the stored index for H will be its maximum index. Otherwise, $\mathrm{sign}(\phi) = -1$, and the stored indices are minimal indices. Therefore, for computation, a filter $\phi : \mathbb{N}^d \to 2^G$ is regarded as

$$\phi = (\mathrm{sign}(\phi), \{(m, H) : 1 \neq H \in \mathrm{im}(\phi)\}).$$

Since all of our filters are series, if $1 \neq H \in \mathrm{im}(\phi)$, define H^+ to be the next term in the descending series. Therefore, $H > H^+$. We first prove a lemma that characterizes when I_H has a maximal element.

Lemma 3. *Suppose $\phi : \mathbb{N}^d \to 2^G$ is a filter and $1 \neq H \in \mathrm{im}(\phi)$. Let $m = \min(I_{H^+})$. The set I_H has a maximum element if, and only if, the dth entry of m is nonzero.*

Proof. Since ϕ is a filter it follows that for all $s \in I_H$, $s \prec m$. Suppose the dth entry of m is 0. Choose $s \in I_H$; then each term in the increasing sequence

$$s \prec s + e_d \prec s + 2e_d \prec \cdots$$

is strictly less than m. Therefore, I_H cannot have a maximum. On the other hand, suppose the dth entry is nonzero. Then $m - e_d \in \mathbb{N}^d$. Since $m - e_d \prec m = \min(I_{H^+})$, it follows that $m - e_d \in I_H$, and thus, $\max(I_H) = m - e_d$. □

Proposition 4. EVALUATE *and* BOUNDARY *are in polynomial time.*

Proof. We first solve Evaluate. Because $\preceq$ is a total order, $\mathbb{N}^d$ is paritioned into intervals I_H and ϕ stores $O(\log|G|)$ pairs (m, H). Thus, given $s \in \mathbb{N}^d$, find the interval I_H such that $s \in I_H$. That is, find consecutive terms in ϕ, (m, H) and (n, K), such that $m \prec s \prec n$. Therefore, Evalute requires $O(\log\log|G|)$ applications of $\preceq$ in $\mathbb{N}^d$.

Because of Lemma 3, Boundary is solved in constant time by the following constructions. If $\text{sign}(\phi) = 1$, then $\partial\phi = (-1, \{(0, G)\} \cup \{(m_H, H^+) : 1 \neq H \in \text{im}(\phi)\})$. If $\text{sign}(\phi) = -1$, then define $n_G = 0$ and, for $1 \neq H \in \text{im}(\phi)$,

$$n_{H^+} = \begin{cases} m_H & \text{if } m_H \text{ exists,} \\ \min(I_{H^+}) & \text{otherwise.} \end{cases}$$

Therefore, $\partial\phi = (-1, \{(n_H, H) : H \in \text{im}(\phi)\})$. □

3.2. *Full filters*

We assume $\phi : \mathbb{N}^d \to 2^G$ is a filter that is not full. We show that we can always fill our filters to make them full.

Lemma 5. *Suppose $\phi : \mathbb{N}^d \to 2^G$ is a filter that is not full. If $H \neq 1$ is the largest subgroup in $\text{im}(\phi)$ such that I_H does not have a maximal element, then there exists a filter $\gamma : \mathbb{N}^d \to 2^G$ with the following properties:*
(1) $\text{im}(\gamma) = \text{im}(\phi)$,
(2) if $K \in \text{im}(\phi)$ such that I_K has a maximum, then $\{s \in \mathbb{N}^d \mid \gamma_s = K\}$ has a maximum, and
(3) $\{s \in \mathbb{N}^d \mid \gamma_s = H\}$ has a maximum.

Proof. Define

$$e = \max(\{m_X + m_Y \mid X, Y \in \text{im}(\phi), X > H, Y > H\} \cap I_H). \tag{3}$$

In case the intersection in (3) is trivial, set $e = \min(I_H)$. Otherwise it is finite and must contain a maximum. Suppose $s, t \in \mathbb{N}^d$ such that $s + t = e$. Suppose $s \in I_X$, $t \in I_Y$ and if s, t are not maximal, then this would contradict (3). Since both s and t are maximal, it follows that

$$[\partial\phi_s, \phi_t] < [\phi_s, \phi_t] \leq \phi_e = H.$$

Hence, for all $s, t \in \mathbb{N}^d$, where $s + t = e$, it follows that $[\partial\phi_s, \phi_t] \leq H^+$.

For each $X \in \text{im}(\phi) - \{H, H^+\}$ define $J_X = I_X$. Additionally, set

$$J_H = \{s \in I_H \mid s \preceq e\} \quad \text{and} \quad J_{H^+} = \{s \in I_H \mid e \prec s\} \cup I_{H^+}.$$

For each $s \in \mathbb{N}^d$, there exists a unique $X \in \text{im}(\phi)$ such that $s \in J_X$, so define $\gamma_s = X$. Since ϕ is a filter, it follows that $\gamma_s \trianglelefteq G$ for all $s \in \mathbb{N}^d$. Moreover, γ is order reversing.

Let $s, t \in \mathbb{N}^d$. There are a few cases depending on whether s, t, or $s+t$ are contained in $\{u \in I_H \mid e \prec u\}$. If only $s + t \in \{u \in I_H \mid e \prec u\}$, then, using (3), $[\gamma_s, \gamma_t] = [\phi_s, \phi_t] < H$. Therefore, $[\gamma_s, \gamma_t] \leq H^+ = \gamma_{s+t}$. All the other cases use similar arguments. Thus, γ is a filter, and the lemma follows. □

Theorem 6. *Suppose $\phi : \mathbb{N}^d \to 2^G$ is a filter. There exists polynomial-time algorithms that*
(1) decide if ϕ is full, and

(2) if ϕ is not full, construct a full filter $\gamma : \mathbb{N}^d \to 2^G$ such that $\operatorname{im}(\gamma) = \operatorname{im}(\phi)$.

Proof. For (1), this follows from Lemma 3. For (2), iterate Lemma 5 until every I_H has a maximal element. This is done with $O(\log^3 |G|)$ operations in $\mathbb{N}^d$ and applications of $\preceq$. □

3.3. Generating filters from prefilters

Next, we generate filters from prefilters $\pi : X \to 2^G$. We store prefilters in the same way we store filters, so in terms of data structures, filters and prefilters are indistinguishable. We note that we only consider the case where $\operatorname{sign}(\pi) = 1$. Because prefilters are not required to satisfy $[\pi_s, \pi_t] \leq \pi_{s+t}$ one can simply change the function of π to allow for maximal indices. However, prefilters often come as refinements of filters, so to keep the filter structure intact, one employs Theorem 6.

We prove a useful lemma first.

Lemma 7. *Let $\pi : X \to 2^G$ be a prefilter with $X = \mathbb{N}^d$. If, for every $1 \neq H \in \operatorname{im}(\pi)$, the set I_H has a maximum, then the same holds for all $1 \neq K \in \operatorname{im}(\overline{\pi})$.*

Proof. Let $H \in \operatorname{im}(\overline{\pi})$ and $s \in \mathbb{N}^d$ such that $\overline{\pi}_s = H$. Since G is finite, there exists a finite and minimal $\mathcal{X} \subset \mathcal{P}_X(s)$ such that

$$H = \prod_{P \in \mathcal{X}} [\pi_P].$$

Let $\mathcal{X}'$ be the set of all partitions $P \in \mathcal{X}$, where each s_i is replaced by its corresponding maximal index. That is, if $P = (s_1, ..., s_k) \in \mathcal{X}$ and $H_i = \pi_{s_i}$, then define $P' = (\max(I_{H_i}))_{i=1}^k$, so $\mathcal{X}' = \{P' : P \in \mathcal{X}\}$. Observe that $|\mathcal{X}| = |\mathcal{X}'|$ and

$$H = \prod_{P \in \mathcal{X}} [\pi_P] = \prod_{P' \in \mathcal{X}'} [\pi_{P'}];$$

however, $\mathcal{X}'$ need not be contained in $\mathcal{P}_X(s)$.

Define

$$t = \min_{(t_1,...,t_k) \in \mathcal{X}'} \left(\sum_{i=1}^{k} t_i \right).$$

Such a minimum exists since $\mathbb{N}^d$ is totally ordered. For $P = (t_1, ..., t_k) \in \mathcal{X}'$, let $u = \sum_i t_i$. Because $t \preceq u$, it follows that $\overline{\pi}_t \geq \overline{\pi}_u$. Therefore, $\overline{\pi}_t = H$.

Let $u \in \mathbb{N}^d$ such that $t \prec u$. Define

$$\mathcal{Y} = \left\{ P \in \mathcal{X}' : u \preceq \sum_i t_i \right\} \quad \text{and} \quad \mathcal{Z} = \left\{ P \in \mathcal{X}' : u \succ \sum_i t_i \right\};$$

note that $\mathcal{Z}$ is nonempty. If

$$H = \prod_{P \in \mathcal{Y}} [\pi_P],$$

then the partitions in $\mathcal{Z}$ are superfluous. Thus, there exists a smaller set of partitions $\mathcal{Y}$, which contradicts the minimality of $\mathcal{X}$. Hence, t is the maximal index for H. □

Now we provide a polynomial-time algorithm for computing filters from prefilters, and hence prove Theorem 1.

Theorem 8. *There exists a polynomial-time algorithm for* GENERATE.

Algorithm. We perform a transitive closure. Start with $\mathcal{S} = \{(m_H, H) : 1 \neq H \in \mathrm{im}(\pi)\}$. We loop through all possible pairs $(x, H), (y, K) \in \mathcal{S}$ such that $[H, K]$ has not previously been computed, selecting pairs where $x + y$ is minimized. Suppose (x, H) and (y, K) are such pairs. Let $s = x + y$, and let $(t, L) \in \mathcal{S}$ such that t is the smallest index with $s \preceq t$. Now we update $\mathcal{S}$; set

$$\mathcal{S} = \begin{cases} (\mathcal{S} - \{(s, L)\}) \cup \{(s, [H, K]L)\} & \text{if } s = t, \\ \mathcal{S} \cup \{(s, [H, K]L)\} & \text{if } s \neq t. \end{cases}$$

Furthermore, for every $(u, X) \in \mathcal{S}$ where $u \prec s$, set

$$\mathcal{S} = (\mathcal{S} - \{(u, X)\}) \cup \{(u, X[H, K]L)\}.$$

At this stage in the loop, it is possible to have duplicate groups in $\mathcal{S}$. We merge them, keeping the largest index, using the order of the subgroup to determine if the groups are equal.

Now we are back to searching for pairs $(x, H), (y, K) \in \mathcal{S}$ such that $[H, K]$ has not been computed previously. If there is another pair, then we remain in the loop. Otherwise, we are done, and we return the filter $(1, \mathcal{S})$. $\square$

Correctness. Let $\phi = \text{GENERATE}(\pi)$. Let $s \in \mathbb{N}^d$ and set $H = \overline{\pi}_s$. There exists a minimal (and finite) $\mathcal{X} \subset \mathcal{P}_{\mathbb{N}^d}(s)$ such that

$$H = \prod_{P \in \mathcal{X}} [\pi_P].$$

If $P = (s_1, ..., s_k) \in \mathcal{X}$, then for $1 < j < k$, we assume $[\pi_{s_1}, ..., \pi_{s_j}] \notin \mathrm{im}(\pi)$. Otherwise, replace P with $(s_1 + \cdots + s_j, s_{j+1}, ..., s_k)$.

At the start of the algorithm, $X = [\pi_{s_1}, \pi_{s_2}]$ has not been computed, so $(s_1 + s_2, X)$ gets inserted into $\mathcal{S}$. Therefore, for $j \geq 2$,

$$(s_{j+1}, \pi_{s_{j+1}}), (s_1 + \cdots + s_j, [\pi_{s_1}, ..., \pi_{s_j}]) \in \mathcal{S},$$

but $[[\pi_{s_1}, ..., \pi_{s_j}], \pi_{s_{j+1}}]$ has not been computed. Thus, insert

$$(s_1 + \cdots + s_{j+1}, [\pi_{s_1}, ..., \pi_{s_{j+1}}])$$

into $\mathcal{S}$. Therefore, $[\pi_P]$ gets computed by the algorithm. Hence, $\mathrm{im}(\phi) = \mathrm{im}(\overline{\pi})$.

Fix $H \in \mathrm{im}(\phi)$. Let $s \in \mathbb{N}^d$ be maximal such that $\phi_s = H$. By Lemma 7, there exists a maximal $t \in \mathbb{N}^d$ such that $\overline{\pi}_t = H$. By the definition of $\overline{\pi}$ from equation (2), it follows that $t \preceq s$. However, for every $P \in \mathcal{P}_{\mathbb{N}^d}(t)$, the algorithm computes the group $[\pi_P]$, so $s \preceq t$. Therefore, $\phi = \overline{\pi}$. $\square$

Timing. Since ϕ is totally ordered, $|\mathrm{im}(\phi)| \leq \log |G|$. Hence, we compute $O(\log^2 |G|)$ commutator subgroups and the orders of $O(\log^2 |G|)$ subgroups. $\square$

3.4. An example

We demonstrate how the algorithm generates a filter from a prefilter. Suppose G is the group of upper unitriangular 5×5 matrices over the finite field $\mathbb{Z}_p$. Let $\gamma : \mathbb{N} \to 2^G$ be the filter obtained from the lower central series of G, where $\gamma_0 = G$. Note that G is generated by $g_i = I_d + E_{i,i+1}$ for $1 \leq i \leq 4$; E_{ij} is the matrix with 1 in the (i,j) entry and 0 elsewhere.

G has a characteristic subgroup $H = \langle g_1, g_4, \gamma_2 \rangle$, where $\gamma_1 > H > \gamma_2$. We construct a prefilter from γ to include H. Define $\pi : \mathbb{N}^2 \to 2^G$ where

$$\pi = \big(1, \big\{\big((1,0), G\big), \big((1,1), H\big), \big((2,0), \gamma_2\big), \big((3,0), \gamma_3\big), \big((4,0), \gamma_4\big)\big\}\big) . \tag{4}$$

Now we want to construct $\phi = \textsc{Generate}(\pi)$. We initialize $\mathcal{S}$ to be the set π given in equation (4). We run through all pairs in $\mathcal{S}$. The first pair, $\big((1,0), G\big)$ and $\big((1,0), G\big)$, provides no new information. The next pair is $\big((1,0), G\big)$ and $\big((1,1), H\big)$. Set $s = (2,1)$, so that $t = (3,0)$. Since $s \neq t$, we include the new subgroup $X = [G,H]\gamma_3$ in $\mathcal{S}$. Thus,

$$\mathcal{S} = \mathcal{S} \cup \big\{\big((2,1), X\big)\big\} .$$

We have no duplicate groups in $\mathcal{S}$ as $X \neq \gamma_i$ for $i \in \{2,3\}$. Therefore, we continue looping through pairs.

The next pair to consider is $\big((1,1), H\big)$ and $\big((1,1), H\big)$. Therefore $s = (2,2)$ and so $t = (3,0)$. Since $s \neq t$, we include the subgroup $Y = [H,H]\gamma_3$ into $\mathcal{S}$, so

$$\mathcal{S} = \mathcal{S} \cup \big\{\big((2,2), Y\big)\big\} .$$

Since $Y = \gamma_3$, we have duplicate groups in $\mathcal{S}$: $\big((2,2), \gamma_3\big)$ and $\big((3,0), \gamma_3\big)$. Because $(2,2) \prec (3,0)$, we remove the entry with $(2,2)$ and only keep the entry with $(3,0)$. Therefore, at this stage,

$$\mathcal{S} = \big\{\big((1,0), G\big), \big((1,1), H\big), \big((2,0), \gamma_2\big), \big((2,1), X\big), \big((3,0), \gamma_3\big), \big((4,0), \gamma_4\big)\big\} .$$

The next pair to consider is $\big((1,0), G\big)$ and $\big((2,0), \gamma_2\big)$ which, again, results in no new information. Computing the commutator of H with γ_2 yields $\big((3,1), \gamma_3\big)$. This gets included in $\mathcal{S}$ because $(3,1)$ is not already included, but $\big((3,0), \gamma_3\big)$ and $\big((3,1), \gamma_3\big)$ are duplicate groups. We remove $\big((3,0), \gamma_3\big)$ from $\mathcal{S}$.

The remaining computations provide no new subgroups, but continue to update the indices. The resulting set is

$$\mathcal{S} = \big\{\big((1,0), G\big), \big((1,1), H\big), \big((2,0), \gamma_2\big), \big((2,1), X\big), \big((3,1), \gamma_3\big), \big((4,2), \gamma_4\big)\big\} .$$

4. Refinements

In (Wilson, 2013, 2015), sources of new subgroups were suggested. We use these in our testing, but we remark that our methods can use any source of refinements, and hence, any prefilter.

Suppose we start with the filter $\eta : \mathbb{N} \to 2^G$ given by the exponent p-central series of G. Then $L(\eta)$ has an associated $\mathbb{N}$-graded Lie algebra, which yields $\mathbb{Z}_p$-bilinear maps from the graded product (e.g.$[,] : L_s \times L_t \rightarrowtail L_{s+t}$). We turn to some associated algebras for these bilinear maps. Suppose $\circ : U \times V \rightarrowtail W$ is a biadditive map of abelian groups;

define the adjoint, centroid, derivation, left scalar, and right scalar rings as

$$\begin{aligned}
\mathrm{Adj}(\circ) &= \{(f,g)\in \mathrm{End}(U)\times \mathrm{End}(V)^{\mathrm{op}} : \forall u\in U, \forall v\in V, (uf)\circ v = u\circ (gv)\},\\
\mathrm{Cent}(\circ) &= \{(f,g,h)\in \mathrm{End}(U)\times \mathrm{End}(V)\times \mathrm{End}(W) : \forall u\in U, \forall v\in V, \forall w\in W,\\
&\qquad (uf)\circ v = u\circ (vg) = (u\circ v)h\},\\
\mathrm{Der}(\circ) &= \{(f,g,h)\in \mathfrak{gl}(U)\times \mathfrak{gl}(V)\times \mathfrak{gl}(W) : \forall u\in U, \forall v\in V, \forall w\in W,\\
&\qquad (uf)\circ v + u\circ (vg) = (u\circ v)h\},\\
\mathcal{L}(\circ) &= \{(f,g)\in \mathrm{End}(U)^{\mathrm{op}}\times \mathrm{End}(W)^{\mathrm{op}} : \forall u\in U, \forall w\in W, (fu)\circ v = g(u\circ v)\}, \text{ and}\\
\mathcal{R}(\circ) &= \{(f,g)\in \mathrm{End}(V)\times \mathrm{End}(W) : \forall v\in V, \forall w\in W, u\circ (vf) = (u\circ v)g\}.
\end{aligned}$$

It is in these nonassociative rings we begin to find more characteristic structure in G (Wilson, 2013, 2015). For example, the Jacobson radical acts on the homogeneous components and yields characteristic subgroups (for $\mathrm{Der}(\circ)$, this is done in the associative enveloping algebra).

5. Examples

Example 1. We consider a p-group found in (Eick et al., 2002, Section 12.1) and refine its lower central series by adding 5 subgroups. Define G by the following power-commutator presentation where the missing commutator relations are assumed to be trivial

$$\begin{aligned}
G = \langle g_1, ..., g_{13} \mid\ & [g_{10}, g_6] = g_{11}, [g_{10}, g_7] = g_{12},\\
& [g_2, g_1] = [g_4, g_3] = [g_6, g_5] = [g_8, g_7] = [g_{10}, g_9] = g_{13}, \text{exponent } p\rangle.
\end{aligned}$$

Let $L(\gamma)$ be the $\mathbb{N}$-graded Lie algebra associated to the lower central series of G. Let $\circ : L_1 \times L_1 \rightarrowtail L_2$ be the nontrivial graded product in $L(\gamma)$. The adjoint algebra A of $\circ$ is a 53 dimensional algebra with a nontrivial Jacobson radical J of dimension 35. Table 1 shows the number of new subgroups added, and Table 2 shows the resulting filter.

i	$\dim(J^i/J^{i+1})$	$\dim(L_1J^i/L_1J^{i+1})$	$[\phi_{(1,i)} : \phi_{(1,i+1)}]$
0	18	1	p
1	18	2	p^2
2	12	4	p^4
3	4	2	p^2
4	1	1	p
5	0	0	1

Table 1. The Jacobson radical of $\mathrm{Adj}(\circ)$ adds four new subgroups.

Example 2. We consider $G = \textsc{SmallGroup}(512,\ 3\,000\,000)$, which is p-class 2 (Eick and O'Brien, 1999). There is a nontrivial derivation refinement which yields three new subgroups. The generation phase produces an additional three more characteristic subgroups, so after one refinement we have eight nontrivial subgroups in the filter. The resulting filter has an adjoint refinement which outputs one new subgroup. The generation phase cannot generate additional subgroups because our prefilter is a composition series, so it just updates the indices.

Maximal Index	Origin of Subgroup	Order	Subgroup
$(1,0)$	G	p^{13}	$\langle g_1, ..., g_{13}\rangle$
$(1,1)$	J	p^{12}	$\langle g_1, ..., g_9, g_{11}, ..., g_{13}\rangle$
$(1,2)$	J^2	p^{10}	$\langle g_1, ..., g_5, g_8, g_9, g_{11}, ..., g_{13}\rangle$
$(1,3)$	J^3	p^6	$\langle g_5, g_8, g_9, g_{11}, ..., g_{13}\rangle$
$(1,4)$	J^4	p^4	$\langle g_9, g_{11}, ..., g_{13}\rangle$
$(2,1)$	G'	p^3	$\langle g_{11}, ..., g_{13}\rangle$
$(2,4)$	generated	p	$\langle g_{13}\rangle$

Table 2. The resulting filter has length 7; nearly a four-fold increase in characteristic subgroups.

Example 3. Let G be a Sylow 2-subgroup of S_{100}; it has order 2^{97} and is p-class 32. Randomly choosing between adjoint and derivation refinements, we run through nine iterations and add 40 new subgroups. Our resulting filter, $\phi : \mathbb{N}^{10} \to 2^G$, is constructed in less than 2 minutes and has 72 nontrivial subgroups.

Example 4. We look at a random sample of 2,000 sections of the Sylow 3-subgroups of classical groups with Lie rank 15. We record how many new subgroups were found, relative to how many we started with, and the time it took to construct these filters. Scatter plots of these data are seen in Figure 1.

6. Closing Remarks

An obvious question might be to consider filters whose monoids are not totally ordered. While there are benefits to this, there are some issues that have to be resolved. For example, the associated Lie ring does not have to have the same order as the group. Indeed, it can be either larger or smaller. Another issue to resolve is the data structure of a filter. Our solution for filters with totally ordered monoids does not readily apply in the general context. One option might be to work with a finite monoid instead of $\mathbb{N}^d$.

A prototype Magma implementation for filters is available from the author.

Acknowledgements

The author is thankful to James B. Wilson and Alexander Hulpke for insightful feedback and encouragement and to the anonymous referee for helpful comments.

References

Bosma, W., Cannon, J., Playoust, C., 1997. The Magma algebra system. I. The user language. J. Symbolic Comput. 24 (3-4), 235–265, computational algebra and number theory (London, 1993).

Cannon, J. J., Holt, D. F., 2003. Automorphism group computation and isomorphism testing in finite groups. J. Symbolic Comput. 35 (3), 241–267.

Eick, B., Leedham-Green, C. R., O'Brien, E. A., 2002. Constructing automorphism groups of p-groups. Comm. Algebra 30 (5), 2271–2295.

Eick, B., O'Brien, E. A., 1999. Enumerating p-groups. J. Austral. Math. Soc. Ser. A 67 (2), 191–205, group theory.

Fitting, H., 1938. Beiträge zur theorie der gruppen endlicher ordnung. Jahresber. Dtsch. Math.-Ver 48, 77–141.

Gorenstein, D., 1980. Finite groups, 2nd Edition. Chelsea Publishing Co., New York.

Holt, D. F., Eick, B., O'Brien, E. A., 2005. Handbook of computational group theory. Discrete Mathematics and its Applications (Boca Raton). Chapman & Hall/CRC, Boca Raton, FL.

Leedham-Green, C. R., Soicher, L. H., 1990. Collection from the left and other strategies. J. Symbolic Comput. 9 (5-6), 665–675, computational group theory, Part 1.

Leedham-Green, C. R., Soicher, L. H., 1998. Symbolic collection using Deep Thought. LMS J. Comput. Math. 1, 9–24 (electronic).

Luks, E. M., 1992. Computing in solvable matrix groups. In: Proc. 33rd IEEE Symposium on Foundations of Computer Science. pp. 111–120.

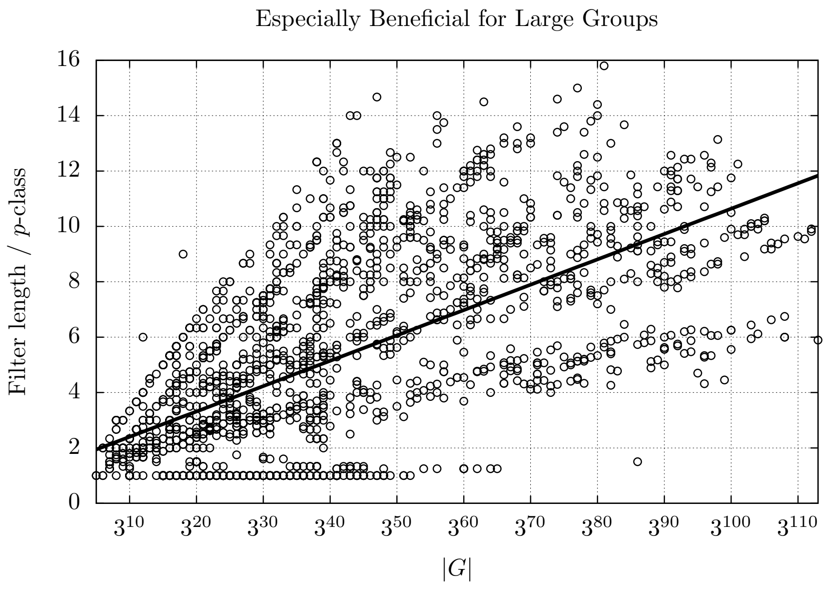
Maglione, J., 2015. Longer nilpotent series for classical unipotent subgroups. J. Group Theory 18 (4), 569–585.

Maglione, J., 2016. Most small p-groups have an involution, in preparation.

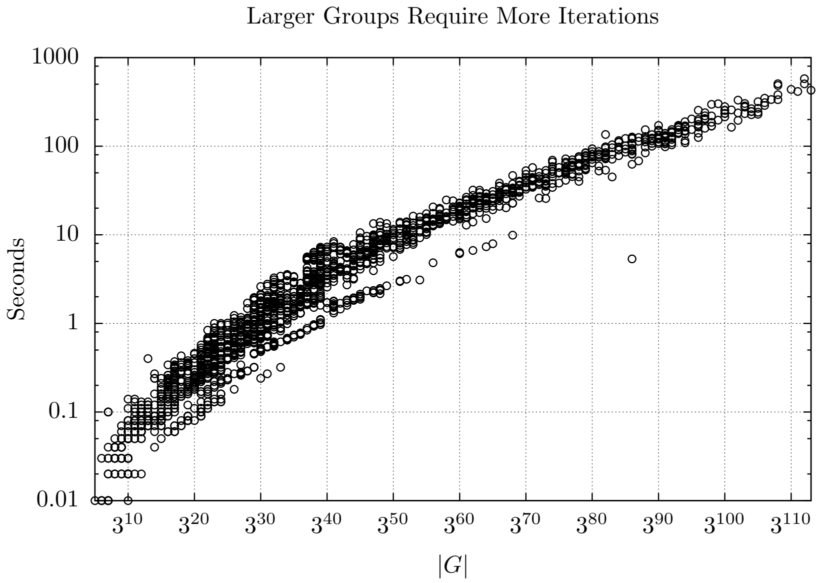
Seress, Á., 2003. Permutation group algorithms. Vol. 152 of Cambridge Tracts in Mathematics. Cambridge University Press, Cambridge.

Wilson, J. B., 2013. More characteristic subgroups, Lie rings, and isomorphism tests for p-groups. J. Group Theory 16 (6), 875–897.

Wilson, J. B., 2015. New lie products for groups and their automorphisms, preprint, `arXiv:1501.04670`.



(a) We report on the growth, relative to the p-class, of each of the refinements. For large groups in this sample, we often find refinements. In addition, we plot the linear regression given by least squares.



(b) We record the total CPU time required to refine the filters. For the larger groups, as many as 20 iterations of Generate were required.

Fig. 1. We sample 2,000 sections of the Sylow 3-subgroups of groups of Lie type. We refine filters until all the algebras in section 4 are semisimple.