Available online at www.sciencedirect.com

ScienceDirect

Journal of Algebra 316 (2007) 828–848

JOURNAL OF Algebra

www.elsevier.com/locate/jalgebra

# Using the Mal'cev correspondence for collection in polycyclic groups

Björn Assmann [*,1], Stephen Linton

*Centre for Interdisciplinary Research in Computational Algebra (CIRCA), University of St. Andrews, North Haugh, St. Andrews, KY16 9SS Fife, Scotland, UK*

Received 6 November 2006

Available online 6 February 2007

Communicated by Eamonn O'Brien

**Abstract**

We describe several approaches for realizing the Mal'cev correspondence between $\mathbb{Q}$-powered nilpotent groups and nilpotent Lie algebras over $\mathbb{Q}$. We apply it to fast collection in polycyclic groups. Our methods are fully implemented and publicly available. We report on the implementation and give runtimes for some example groups.
© 2007 Elsevier Inc. All rights reserved.

*Keywords:* Polycyclically presented groups; Mal'cev correspondence; Collection; Collection from the left

## 1. Introduction

Logarithms were invented by John Napier (1550–1617) in order to speed up computations with big numbers. His idea was to reduce multiplication and division to simple addition and subtraction by using transformations contained in a logarithm table.

This paper is about how logarithms, or more generally the connection between groups and Lie algebras, can be used to speed up multiplications in a non-commutative setting, namely for infinite polycyclic groups.

[*] Corresponding author.
*E-mail addresses:* bjoern@dcs.st-and.ac.uk (B. Assmann), sal@dcs.st-and.ac.uk (S. Linton).
[1] The first author was supported by a PhD fellowship of the "Gottlieb Daimler- und Karl Benz-Stiftung" and the UK Engineering and Physical Science Research Council (EPSRC).

0021-8693/$ – see front matter © 2007 Elsevier Inc. All rights reserved.
doi:10.1016/j.jalgebra.2007.01.028

The class of polycyclic groups has been shown to be very fruitful for investigations, both from a theoretical and a computational point of view [7,10,21,22]. Polycyclic groups can be represented very efficiently on a computer via consistent polycyclic presentations. Every element of a polycyclically presented group has a unique normal form. An algorithm for computing this normal form is called a collection algorithm. Such an algorithm lies at the heart of most methods dealing with polycyclically presented groups. The current state of the art is "collection from the left" [8,13,24].

We describe a method, which we call Mal'cev collection, that in some cases is dramatically faster than collection from the left, while using less memory.

The part of the logarithm table in our method is played by the Mal'cev correspondence. This is a one-to-one correspondence between $\mathbb{Q}$-powered nilpotent groups and nilpotent Lie algebras over $\mathbb{Q}$. It was discovered by Anatoly Mal'cev in 1951 [16]. We recall some of the important features of this correspondence and describe several approaches for its realization on a computer.

Our methods are fully implemented in the computer algebra system GAP [23] and publicly available as part of the GAP package Guarana [2]. A report on the implementation including runtimes for some example groups is given at the end of this paper.

## 2. Polycyclic presentations

In this section we recall some of the basic properties of polycyclic presentations. For more information we refer to [10,22] or [7].

Let $G$ be a polycyclic group. A *polycyclic sequence* of $G$ is a sequence of elements $\mathcal{G} = (g_1, \ldots, g_n)$ of $G$ such that the subgroups $G_i = \langle g_i, \ldots, g_n \rangle$ form a subnormal series $G = G_1 > \cdots > G_n > G_{n+1} = \{1\}$ with non-trivial cyclic factors.

Let $r_i = [G_i : G_{i+1}]$ and $I = \{i \in \{1, \ldots, n\} \mid r_i < \infty\}$. Then each element $g \in G$ has a unique normal form with respect to the polycyclic sequence: $\mathrm{nf}(g) = g_1^{e_1} \cdots g_n^{e_n}$ with $e_i \in \mathbb{Z}$ and $0 \leqslant e_i < r_i$ for $i \in I$. Thus $g$ can be represented by the exponent vector $(e_1, \ldots, e_n)$ with respect to $\mathcal{G}$. For a given exponent vector $e = (e_1, \ldots, e_n)$ we denote by $g^e$ the element $g_1^{e_1} \cdots g_n^{e_n}$.

Each polycyclic sequence $(g_1, \ldots, g_n)$ of $G$ defines a presentation of $G$ on the generators $g_1, \ldots, g_n$ with relations of the form

$$\begin{aligned} g_i^{g_j} &= g_{j+1}^{e(i,j,j+1)} \cdots g_n^{e(i,j,n)} \quad \text{for } 1 \leqslant j < i \leqslant n, \\ g_i^{g_j^{-1}} &= g_{j+1}^{f(i,j,j+1)} \cdots g_n^{f(i,j,n)} \quad \text{for } 1 \leqslant j < i \leqslant n, \\ g_i^{r_i} &= g_{i+1}^{\ell(i,i+1)} \cdots g_n^{\ell(i,n)} \quad \text{for } i \in I, \end{aligned}$$

where the right-hand sides in these relations are the normal forms of the elements on the left-hand sides. It is well known that these relations define $G$. Such a presentation is called a *consistent polycyclic presentation* for $G$. The term consistent refers to the fact that every element in this finitely presented group on the abstract generators $g_1, \ldots, g_n$ has a unique normal form $g_1^{e_1} \cdots g_n^{e_n}$ with $e_i \in \mathbb{Z}$ and $0 \leqslant e_i < r_i$ for $i \in I$. Throughout this paper all polycyclic presentations are consistent.

Let $H$ be a $\mathcal{T}$-group, i.e. a finitely generated torsion-free nilpotent group. A polycyclic sequence $(h_1, \ldots, h_n)$ is called a *Mal'cev basis* for $H$ if the subgroups $\langle h_i, \ldots, h_n \rangle$ form a central series with infinite cyclic factors. Since the upper central series of a $\mathcal{T}$-group has torsion-free factors, every $\mathcal{T}$-group has a Mal'cev basis.

## 3. Mal'cev correspondence

A group $G$ is said to be $\mathbb{Q}$-*powered* if for every $g \in G, n \in \mathbb{N}$ there exists a unique $h \in G$ such that $h^n = g$. We denote this element $h$ by $g^{\frac{1}{n}}$ and write $g^{\frac{p}{q}}$ for $(g^{\frac{1}{q}})^p$ where $p \in \mathbb{Z}, q \in \mathbb{N}$. Note that a $\mathbb{Q}$-powered group has to be torsion-free. The Mal'cev correspondence [16] is a one-to-one correspondence between $\mathbb{Q}$-powered nilpotent groups and nilpotent Lie algebras over $\mathbb{Q}$. In this section we recall some well-known properties of this connection. For more background we refer to [21, Chapter 6], [12, Chapter 9,10] and [3, Chapter 4].

Let $L$ be a nilpotent Lie algebra over $\mathbb{Q}$ and let $x, y \in L$. Using the Baker–Campbell–Hausdorff formula [12, Definition 9.6] $H(x, y) = x + y + \frac{1}{2}[x, y]_L + \cdots$, where we denote by $[\,,\,]_L$ the Lie commutator in $L$, we can define the operations of a $\mathbb{Q}$-powered group on $L$ by

$$x * y = H(x, y), \tag{1}$$

$$x^q = qx \quad \text{for } q \in \mathbb{Q}. \tag{2}$$

Conversely, let $G$ be a $\mathbb{Q}$-powered nilpotent group and let $g, h \in G$. Then using the inverse Baker–Campbell–Hausdorff formulae [12, Lemma 10.7]

$$h_1(g, h) = gh[g, h]_G{}^{-\frac{1}{2}} \cdots,$$
$$h_2(g, h) = [g, h]_G[h, g, h]_G{}^{\frac{1}{2}}[h, g, g]_G{}^{\frac{1}{2}} \cdots,$$

where we denote by $[\,,\,]_G$ the group commutator in $G$, we can define the operations of a rational Lie algebra on $G$ by

$$g + h = h_1(g, h), \tag{3}$$

$$[g, h]_L = h_2(g, h), \tag{4}$$

$$qg = g^q \quad \text{for } q \in \mathbb{Q}. \tag{5}$$

**Theorem 3.1.** *For every $\mathbb{Q}$-powered nilpotent group $G$, the corresponding rational nilpotent Lie algebra $L_G$ is defined on the same underlying set $L_G = G$, with Lie $\mathbb{Q}$-algebra operations* (3)–(5). *Conversely, for every rational nilpotent Lie algebra $L$, the corresponding $\mathbb{Q}$-powered nilpotent group $G_L$ is defined on the same underlying set $G_L = L$, with group operations* (1), (2) *of a $\mathbb{Q}$-powered group. These transformations are inverses of one another*: *$L_{G_L} = L$ as rational Lie algebras* (*that is, not only sets, but all operations coincide*), *and, similarly, $G_{L_G} = G$ as $\mathbb{Q}$-powered groups.*

**Proof.** See [12, Theorem 10.11]. □

Let $G$ be a $\mathbb{Q}$-powered nilpotent group. To avoid confusion between $G$ and $L_G$ in the following, we will denote by $\operatorname{Log}(g)$ the element of $L_G$ which corresponds to $g \in G$;

$$G \ni g \quad \leftrightarrow \quad \operatorname{Log}(g) \in L_G = \operatorname{Log}(G) = \{\operatorname{Log}(g) \mid g \in G\}.$$

For a rational nilpotent Lie algebra $L$, we will denote by $\operatorname{Exp}(x)$, the element of $G_L$ which corresponds to $x \in L$. This notation is motivated by the fact that for $\mathbb{Q}$-powered subgroups of

$\mathrm{Tr}_1(n,\mathbb{Q})$ the Lie algebra $L_G$ is isomorphic to $\log(G)$, where log is the usual logarithm map [21]. Distinguishing $G$ and $L_G$ in this way, in the following it will be clear from the context if we mean by $[\,,\,]$ the Lie commutator or the group commutator. Log can be regarded as a mapping between $G$ and $\mathrm{Log}(G)$. For $g,h \in G$ and $q \in \mathbb{Q}$ we have

$$\begin{aligned}\mathrm{Log}(gh) &= \mathrm{Log}(g) * \mathrm{Log}(h),\\ \mathrm{Log}\big(h_1(g,h)\big) &= \mathrm{Log}(g) + \mathrm{Log}(h),\\ \mathrm{Log}\big(h_2(g,h)\big) &= \big[\mathrm{Log}(g),\mathrm{Log}(h)\big],\\ \mathrm{Log}\big(g^q\big) &= q\mathrm{Log}(g).\end{aligned}$$

Similarly Exp can be regarded as a mapping between $L$ and $\mathrm{Exp}(L)$.

Let $H$ be a torsion-free nilpotent group. A $\mathbb{Q}$-powered group $\hat{H}$, containing $H$, is said to be a *$\mathbb{Q}$-powered hull* of $H$, if for every element $h \in \hat{H}$ there exists $z \in \mathbb{N}$ such that $h^z \in H$.

**Theorem 3.2.** *Let $H$ be a torsion-free nilpotent group. Then $H$ has a $\mathbb{Q}$-powered hull $\hat{H}$ of the same nilpotency class and $\hat{H}$ is unique up to isomorphism. Every automorphism of $H$ extends to an automorphism of $\hat{H}$. If $H \leqslant G$ and $G$ is $\mathbb{Q}$-powered, then $G$ contains a $\mathbb{Q}$-powered hull of $H$.*

**Proof.** See [12, Corollary 9.19 and Theorem 9.20]. □

Given the fact that all $\mathbb{Q}$-powered hulls of a $\mathcal{T}$-group $H$ are isomorphic, we will identify them in the following and speak of the $\mathbb{Q}$-powered hull $\hat{H}$ of $H$. The Lie algebra $L_{\hat{H}}$ corresponding to $\hat{H}$ will be denoted by $\mathcal{L}(H)$. Note that $\mathcal{L}(H)$ is spanned by $\mathrm{Log}(H) \subset \mathcal{L}(H)$ over the rationals, because every element $h \in \hat{H}$ has some power $h^z$ lying in $H$; thus $\mathrm{Log}(h) = \frac{1}{z}\mathrm{Log}(h^z) \in \mathbb{Q}\mathrm{Log}(H)$.

In Section 5, we will use the Mal'cev correspondence for computations with automorphisms of a $\mathcal{T}$-group $H$. The next theorem shows that the automorphisms of $\hat{H}$ and $\mathcal{L}(H)$ are in one-one correspondence; it is a direct consequence of the fact that the Lie algebra operations in $\mathcal{L}(H)$ can be defined in terms of the group operations of $\hat{H}$ and vice-versa.

**Theorem 3.3.** *Let $G$ be a $\mathbb{Q}$-powered nilpotent group and $L$ its corresponding Lie algebra. Then the map* $\tilde{}: \mathrm{Aut}(G) \to \mathrm{Aut}(L)$, *defined by* $\varphi \mapsto \mathrm{Exp} \circ \varphi \circ \mathrm{Log}$ *is an isomorphism.*

**Proof.** Let $\varphi \in \mathrm{Aut}(G)$ and $g,h \in G$. Then

$$\begin{aligned}\big(\mathrm{Log}(g)+\mathrm{Log}(h)\big)^{\tilde{\varphi}} &= \mathrm{Log}\big(h_1(g,h)\big)^{\tilde{\varphi}}\\ &= \mathrm{Log}\big(h_1(g,h)^{\varphi}\big)\\ &= \mathrm{Log}\big(h_1\big(g^{\varphi},h^{\varphi}\big)\big)\\ &= \mathrm{Log}\big(g^{\varphi}\big)+\mathrm{Log}\big(h^{\varphi}\big)\\ &= \mathrm{Log}(g)^{\tilde{\varphi}}+\mathrm{Log}(h)^{\tilde{\varphi}}.\end{aligned}$$

With a similar argument we see that $[\mathrm{Log}(g),\mathrm{Log}(h)]^{\tilde{\varphi}} = [\mathrm{Log}(g)^{\tilde{\varphi}},\mathrm{Log}(h)^{\tilde{\varphi}}]$ and that for $q \in \mathbb{Q}$ we have $(q\mathrm{Log}(g))^{\tilde{\varphi}} = q(\mathrm{Log}(g))^{\tilde{\varphi}}$. Thus $\tilde{\varphi} \in \mathrm{Aut}(L)$.

Similarly we see that for $\tau \in \mathrm{Aut}(L)$, the mapping $\mathrm{Log} \circ \tau \circ \mathrm{Exp}$ is in $\mathrm{Aut}(G)$. Therefore, since $\varphi \mapsto \mathrm{Exp} \circ \varphi \circ \mathrm{Log}$ and $\tau \mapsto \mathrm{Log} \circ \tau \circ \mathrm{Exp}$ are inverses of each other, $\tilde{}$ is a bijection between $\mathrm{Aut}(G)$ and $\mathrm{Aut}(L)$. Further for $\varphi, \psi \in \mathrm{Aut}(G)$ we have that $\widetilde{\varphi\psi} = \tilde{\varphi}\tilde{\psi}$ and thus $\tilde{}$ is an isomorphism. □

The next theorem explains how $\mathrm{Aut}(H)$ fits into the relationship between the automorphism group of $\hat{H}$ and $\mathcal{L}(H)$.

**Theorem 3.4.** *Let $H$ be a torsion-free nilpotent group and $\hat{H}$ its $\mathbb{Q}$-powered hull. Let $\Gamma$ be the stabilizer in* $\mathrm{Aut}(\mathcal{L}(H))$ *of the set* $\mathrm{Log}(H)$. *Then $\varphi \in \mathrm{Aut}(\hat{H})$ is an automorphism of $H$, i.e. $H^{\varphi} = H$, if and only if $\tilde{\varphi} \in \Gamma$.*

**Proof.** Let $\varphi \in \mathrm{Aut}(\hat{H})$ and assume that $H^{\varphi} = H$. Then we have $\mathrm{Log}(H)^{\tilde{\varphi}} = \mathrm{Log}(H^{\varphi}) = \mathrm{Log}(H)$. Thus $\tilde{\varphi} \in \Gamma$. Conversely assume that $\tilde{\varphi} \in \Gamma$. Then $H^{\varphi} = \mathrm{Exp}(\mathrm{Log}(H)^{\tilde{\varphi}}) = H$ and thus $H^{\varphi} = H$. □

**Lemma 3.5.** *Let $(g_1, \ldots, g_l)$ be a Mal'cev basis for a $\mathcal{T}$-group $H$. Then $\mathcal{B} = \{\mathrm{Log}(g_1), \ldots, \mathrm{Log}(g_l)\}$ is a basis for the Lie algebra $\mathcal{L}(H)$. In particular, the dimension of $\mathcal{L}(H)$ is equal to the Hirsch length of $H$.*

**Proof.** Let $g = g_1^{a_1} \cdots g_l^{a_l} \in H$. Then $\mathrm{Log}(g) = a_1\mathrm{Log}(g_1) * \cdots * a_l\mathrm{Log}(g_l)$. Thus, it is sufficient to show that $\mathcal{B}$ is basis for the Lie algebra $L$ generated by $\mathcal{B}$. We show this via induction over $l$, the Hirsch length of $H$.

If $H = \langle g_1 \rangle$ then $\{\mathrm{Log}(g_1)\}$ is a basis for $\mathcal{L}(H)$. Assume that the lemma is true for all $\mathcal{T}$-groups of Hirsch length $l - 1$. First we show that $\mathcal{B}$ is a generating set for $\mathcal{L}(H)$. By assumption the vector spaces $\langle \mathrm{Log}(g_2), \ldots, \mathrm{Log}(g_l) \rangle_{\mathbb{Q}}$ and $\langle \mathrm{Log}(g_1), \mathrm{Log}(g_3), \ldots, \mathrm{Log}(g_l) \rangle_{\mathbb{Q}}$ are closed under taking Lie brackets. Thus we have to show that $[\mathrm{Log}(g_1), \mathrm{Log}(g_2)] \in \langle \mathcal{B} \rangle_{\mathbb{Q}}$. By Corollary 3 of [21, Chapter 6] we have that $[\mathrm{Log}(g_1), \mathrm{Log}(g_2)] = \mathrm{Log}([g_1, g_2]) + \sum_i \alpha_i \mathrm{Log}(\chi_i(g_1, g_2))$, where $\alpha_i \in \mathbb{Q}$ and $\chi_i$ is a repeated group theoretic commutator in $g_1, g_2$ of length $\geqslant 3$. Since $[g_1, g_2], \chi_i(g_1, g_2) \in \langle g_3, \ldots, g_l \rangle$ the right-hand side of the last equation is contained in the Lie algebra $\mathcal{L}(\langle g_3, \ldots, g_l \rangle)$ and thus in the $\mathbb{Q}$-vector space spanned by $\mathcal{B}$.

It remains to show that the elements of $\mathcal{B}$ are linearly independent. By induction hypothesis $\mathrm{Log}(g_2), \ldots, \mathrm{Log}(g_l)$ are linearly independent. So assume that $\mathrm{Log}(g_1) \in \langle \mathrm{Log}(g_2), \ldots, \mathrm{Log}(g_l) \rangle_{\mathbb{Q}} = L_2$. Therefore $g_1 \in \mathrm{Exp}(L_2)$ which is equal to the $\mathbb{Q}$-powered hull of $\langle g_2, \ldots, g_l \rangle$. Thus there must be an $m \in \mathbb{N}$ such that $g_1^m \in \langle g_2, \ldots, g_l \rangle$. Since $(g_1, \ldots, g_l)$ is a Mal'cev basis this is a contradiction. □

## 4. Setup for computing the correspondence

In this section we show how the Mal'cev correspondence between the radicable hull of a $\mathcal{T}$-group $G$ and the Lie algebra $\mathcal{L}(G)$ can be set up on a computer. We assume that $G$ is given by a polycyclic presentation with respect to a Mal'cev basis $\mathcal{G} = (g_1, \ldots, g_l)$. Note that $\mathcal{B} = \{\mathrm{Log}(g_1), \ldots, \mathrm{Log}(g_l)\}$ is a basis of $\mathcal{L}(G)$. We show how to solve the following three tasks:

- **Lie algebra presentation:** Determine a computer presentation of $\mathcal{L}(G)$.
- **Logarithm:** Given an element $g = g_1^{e_1} \cdots g_l^{e_l} \in \hat{G}$, compute the coefficient vector $(\alpha_1, \ldots, \alpha_l)$ such that $\mathrm{Log}(g) = \sum_{i=1}^{l} \alpha_i \mathrm{Log}(g_i)$.

- **Exponential:** Given an element $x = \sum_{i=1}^{l} \alpha_i \mathrm{Log}(g_i) \in \mathcal{L}(G)$, compute the exponent vector $(e_1, \dots, e_l)$ such that $\mathrm{Exp}(x) = g_1^{e_1} \cdots g_l^{e_l}$.

We present two approaches for solving these tasks. The first uses the fact that every $\mathcal{T}$-group can be embedded in an upper unitriangular matrix group and is discussed in Section 4.1. The second makes use of the Baker–Campbell–Hausdorff formula and related identities, see Section 4.2. Further we present in Section 4.3 a symbolic approach for computing logarithms and exponentials. Finally, we compare these methods and report on their implementations in Section 4.4.

## *4.1. Via matrix embeddings*

This method uses a vector space of matrices to represent $\mathcal{L}(G)$ on a computer. For more background on this approach we refer to [21, Chapter 6] and for a more detailed description of its algorithmic realization to [1].

Every $\mathcal{T}$-group $G$ has a faithful matrix representation $\beta : G \to \mathrm{Tr}_1(n, \mathbb{Q})$ for some $n \in \mathbb{N}$, where $\mathrm{Tr}_1(n, \mathbb{Q})$ is the set of all rational upper unitriangular matrices of degree $n$ [21, Chapter 3]. Denote by log, respectively exp, the usual logarithm, respectively exponential map, defined by the usual power series, and let $\mathrm{Tr}_0(n, \mathbb{Q})$ be the set of all rational upper triangular matrices with 0s on the diagonal. Note that log and exp are in fact given by polynomials, because only the first $n$ terms of their powers series are non-zero on $\mathrm{Tr}_1(n, \mathbb{Q})$, respectively $\mathrm{Tr}_0(n, \mathbb{Q})$.

**Theorem 4.1.** *The maps* log *and* exp *are mutually inverse bijections between* $\mathrm{Tr}_1(n, \mathbb{Q})$ *and* $\mathrm{Tr}_0(n, \mathbb{Q})$ *and furthermore* $L = \mathbb{Q}\log(G\beta)$*, i.e. the* $\mathbb{Q}$*-span of* $\{\log(g\beta) \mid g \in G\}$ *in* $\mathrm{Tr}_0(n, \mathbb{Q})$*, is such that* $\exp L \cong (L, *)$ *is a* $\mathbb{Q}$*-powered hull of* $G\beta$*.*

**Proof.** See [21, Chapter 6]. □

By Theorem 3.2, $(L, *)$ and $(\mathcal{L}(G), *)$ are isomorphic as $\mathbb{Q}$-powered nilpotent groups and thus, by the Mal'cev correspondence, $L$ and $\mathcal{L}(G)$ are isomorphic as rational Lie algebras.

For a $\mathcal{T}$-group $G$, given by a polycyclic presentation, it is possible to compute a faithful representation $\beta : G \to \mathrm{Tr}_1(n, \mathbb{Q})$ [5,15,19]. An implementation of [5] is publicly available as part of the computer algebra system GAP [23]. In order to be able to go back and forth between $G$ and $G\beta$, it is necessary to compute a *constructive* polycyclic sequence for $G\beta$. This is a polycyclic sequence $\mathcal{M} = (M_1, \dots, M_l)$ for $G\beta$ such that there exists a practical algorithm, which, given any $h \in G\beta$, determines the normal form $\mathrm{nf}(h)$ with respect to $\mathcal{M}$. It is well known how to compute a constructive polycyclic sequence for a given finitely generated subgroup of $\mathrm{Tr}_1(n, \mathbb{Q})$ which is also a Mal'cev basis of $H$, see for example [22, Chapter 9]. By changing the underlying generating set of the polycyclic presentation of $G$, we can assume in the following that $\mathcal{G} = (g_1, \dots, g_l)$ is a pc-sequence of $G$ such that $(g_1\beta, \dots, g_l\beta)$ is constructive pc-sequence for $G\beta$. As a basis for $\mathcal{L}(G) \cong \mathbb{Q}\log(G\beta)$ we use the set of matrices $(\log(g_1\beta), \dots, \log(g_l\beta))$.

The task Logarithm can be solved as follows. Given $g = g_1^{e_1} \cdots g_l^{e_l} \in \hat{G}$, we compute $M = (g_1\beta)^{e_1} \cdots (g_l\beta)^{e_l}$ and then $\log(M)$. Finally, by solving linear equations, we determine $(\alpha_1, \dots, \alpha_l)$ such that $\sum \alpha_i \log(g_i\beta) = \log(M)$.

For the task Exponential we do the following. Given $x = \sum_{i=1}^{l} \alpha_i \log(g_i\beta)$, we compute $\exp(x)$. Then we use the constructive pc-sequence $(g_1\beta, \dots, g_l\beta)$ of $G\beta$ to compute the exponent vector $(e_1, \dots, e_l)$ of $\exp(x)$. The algorithm to compute $(e_1, \dots, e_l)$ is a straightforward generalization of the method in [22, Chapter 9].

**Example 4.2.** Let $G = F_{2,2}$ be the free nilpotent of class two group on two generators. Then

$$G = \langle g_1, g_2, g_3 \mid g_2^{(g_1^{\pm 1})} = g_2 g_3^{\pm 1} \rangle$$

is a polycyclic presentation for $G$ and $\mathcal{G} = (g_1, g_2, g_3)$ is a Mal'cev basis. The embedding $\beta \to \mathrm{Tr}_1(3, \mathbb{Q})$, as computed by the algorithm in [5], is given by

$$g_1\beta = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}, \qquad g_2\beta = \begin{pmatrix} 1 & -1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \qquad g_3\beta = \begin{pmatrix} 1 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

In this example $(g_1\beta, g_2\beta, g_3\beta)$ is a constructive polycyclic sequence for $G\beta$ and therefore we do not have to change the underlying generating set of $G$ and set $\mathcal{M} = (g_1\beta, g_2\beta, g_3\beta)$. The corresponding basis of $\mathbb{Q}\log(G\beta) \cong \mathcal{L}(G)$ consists of

$$\log(M_1) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \log(M_2) = \begin{pmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \log(M_3) = \begin{pmatrix} 0 & 0 & -1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

Let $x = \log(M_1) + \log(M_2) + \log(M_3) \in \mathcal{L}(G)$ and assume that we want to compute the exponent vector of $\mathrm{Exp}(x)$, i.e. the vector $(e_1, e_2, e_3)$ such that

$$\left(g_1^{e_1} g_2^{e_2} g_3^{e_3}\right)\beta = \exp(x).$$

First we compute

$$\exp(x) = \frac{x^0}{0!} + \frac{x^1}{1!} + \frac{x^2}{2!} = \begin{pmatrix} 1 & -1 & -3/2 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}.$$

Now we can read off from the first subdiagonal that $e_1 = 1$ and $e_2 = 1$. Next we divide off and compute

$$(g_3\beta)^{e_3} = \left((g_1\beta)^{e_1}(g_2\beta)^{e_2}\right)^{-1} \exp(x) = \begin{pmatrix} 1 & 0 & -3/2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

This implies $e_3 = 3/2$.

## *4.2. Via the Baker–Campbell–Hausdorff formula*

This method uses an abstract vector space and a structure constant table with respect to the basis $\mathcal{B}$ to present $\mathcal{L}(G)$ on a computer.

For the computation of the structure constants, we use a certain identity $\mathcal{I}$ in $F_{2,c}$, the free nilpotent of class $c$ group on 2 generators; it expresses the Lie bracket of elements in $\mathrm{Log}(F_{2,c})$ in terms of a linear combination of logarithms of group commutators. For example in $F_{2,3}$ this identity $\mathcal{I}$ is

$$\left[\mathrm{Log}(e), \mathrm{Log}(f)\right] = \mathrm{Log}\left([e, f]\right) + \frac{1}{2}\mathrm{Log}\left([f, e, f]\right) + \frac{1}{2}\mathrm{Log}\left([f, e, e]\right).$$

It can be obtained as follows. It is well known that for any commutator $\kappa$ of weight $n \geqslant 2$,

$$\kappa\big(\mathrm{Log}(e), \mathrm{Log}(f)\big) = \mathrm{Log}\big(\kappa(e, f)\big) + T \tag{6}$$

where $\kappa(\mathrm{Log}(e), \mathrm{Log}(f))$ is the Lie commutator in $\mathrm{Log}(e), \mathrm{Log}(f) \in \mathcal{L}(F_{2,c})$, $\kappa(e, f)$ is the group commutator in $e, f \in F_{2,c}$ and $T$ is a linear combination of Lie commutators of length $\geqslant n + 1$ [12, Lemma 9.1]. The terms of the right-hand side of (6) can be determined using a straight forward generalization of the method in [20, §8] and the Dynkin bracket operator defined in [25, Chapter 2]. For $\kappa(e, f) = [e, f]$ we get

$$\begin{aligned}\big[\mathrm{Log}(e), \mathrm{Log}(f)\big] = \mathrm{Log}\big([e, f]\big) &+ \frac{1}{2}\big[\mathrm{Log}(f), \mathrm{Log}(e), \mathrm{Log}(f)\big] \\ &+ \frac{1}{2}\big[\mathrm{Log}(f), \mathrm{Log}(e), \mathrm{Log}(e)\big] + \cdots.\end{aligned} \tag{7}$$

Now we replace all occurrences of Lie commutators of weight $\geqslant 3$ in (7) with the corresponding right-hand site of (6) and continue until no Lie bracket of weight $\geqslant 3$ appears in the equation. This process must stop, because $F_{2,c}$ has finite nilpotency class and thus (6) has only finitely many terms, and because a replacement of a Lie commutator of weight $k$ only introduces Lie commutators of weight $\geqslant k + 1$.

Let $\mathcal{G} = (g_1, \dots, g_l)$ be a Mal'cev basis of $G$ and denote by $\mathcal{B}$ be the corresponding basis $\{\mathrm{Log}(g_1), \dots, \mathrm{Log}(g_l)\}$ of $\mathcal{L}(G)$. We show by induction on $l$ that it is possible to compute the structure constant table of the Lie algebra $\mathcal{L}(G)$ with respect to $\mathcal{B}$.

If $l = 1$, then $\mathcal{L}(G)$ is abelian and so the structure constant table of $\mathcal{B}$ is known. If $l > 1$, then we can assume by induction that the structure constant table of $\{\mathrm{Log}(g_2), \dots, \mathrm{Log}(g_l)\}$ is already known and that we can compute $\mathrm{Log}(g)$ for $g$ in the $\mathbb{Q}$-powered hull of $\langle g_2, \dots, g_l\rangle$. Then using $\mathcal{I}$, we can express $[\mathrm{Log}(g_1), \mathrm{Log}(g_i)]$ as a linear combination of logarithms of groups commutators $\kappa(g_1, g_i)$. Since $\kappa(g_1, g_i)$ is in $\langle g_2, \dots, g_l\rangle$ for any commutator $\kappa$, we can compute $\mathrm{Log}(\kappa(g_1, g_i))$ and therefore determine the coefficients of $[\mathrm{Log}(g_1), \mathrm{Log}(g_i)]$ with respect to $\mathcal{B}$. Thus we can compute the structure constant table of $\mathcal{B}$.

For the computation of Logarithms we use the fact that $\mathrm{Log}(gh) = \mathrm{Log}(g) * \mathrm{Log}(h)$. Thus for given $g = g_1^{e_1} \cdots g_l^{e_l} \in \hat{G}$ we have that $\mathrm{Log}(g) = (e_1\mathrm{Log}(g_1)) * \cdots * (e_l\mathrm{Log}(g_l))$, and therefore the coefficients of $\mathrm{Log}(g) = \sum \alpha_i \mathrm{Log}(g_i)$ can be computed by using the Baker–Campbell–Hausdorff formula and the structure constant table of $\mathcal{B}$.

It remains to solve the task Exponential. For a given element $x = \sum_{i=1}^{l} \alpha_i \mathrm{Log}(g_i) \in \mathcal{L}(G)$ we have that $x = (\alpha_1\mathrm{Log}(g_1)) * (-\alpha_1\mathrm{Log}(g_1)) * x$ and $y = (-\alpha_1\mathrm{Log}(g_1)) * x \in \langle \mathrm{Log}(g_2), \dots, \mathrm{Log}(g_l)\rangle_{\mathbb{Q}}$. Thus $e_1 = \alpha_1$. Using the structure constant table of $\mathcal{B}$ and the BCH-formula we can compute $y = (-\alpha_1\mathrm{Log}(g_1)) * x$. By induction on $l$, we can assume that we can determine $f_2, \dots, f_l$ such that $g_2^{f_2} \cdots g_l^{f_l} = \mathrm{Exp}(y)$. Since $\mathrm{Exp}(x) = g_1^{\alpha_1}\mathrm{Exp}(y)$ we deduce that $(e_1, \dots, e_l) = (\alpha_1, f_2, \dots, f_l)$.

**Example 4.3.** Let $G = F_{2,2}$ be given as in Example 4.2. The identity $\mathcal{I}$ in $F_{2,2}$ is $[\mathrm{Log}(e), \mathrm{Log}(f)] = \mathrm{Log}([e, f])$ for $e, f \in G$. Therefore $[\mathrm{Log}(g_2), \mathrm{Log}(g_1)] = \mathrm{Log}(g_3)$ and $[\mathrm{Log}(g_3), \mathrm{Log}(g_1)] = [\mathrm{Log}(g_3), \mathrm{Log}(g_2)] = 0$.

Let $g = g_1 g_2^5$ and suppose that we want to compute the coefficients of $\mathrm{Log}(g)$. We have that $\mathrm{Log}(g) = \mathrm{Log}(g_1) * \mathrm{Log}(g_2^5) = \mathrm{Log}(g_1) + 5\mathrm{Log}(g_2) + \frac{1}{2}[\mathrm{Log}(g_1), 5\mathrm{Log}(g_2)]$. Thus $\mathrm{Log}(g) = \mathrm{Log}(g_1) + 5\mathrm{Log}(g_2) - \frac{5}{2}\mathrm{Log}(g_3)$.

## 4.3. *Symbolic* Log *and* Exp

It is well known that, in the context of $\mathcal{T}$-groups, Log and Exp can be described by polynomial functions [11, Chapter 6]. In this section we show how to compute these functions and apply them for the computations of logarithms and exponentials.

**Lemma 4.4.** *Let $G$ be a $\mathcal{T}$-group with Mal'cev basis $\mathcal{G} = (g_1, \ldots, g_l)$.*

(i) *Define $l$ functions $\bar{\alpha}_1, \ldots, \bar{\alpha}_l$ in $l$ rational variables such that*

$$\sum_{i=1}^{l} \bar{\alpha}_i \mathrm{Log}(g_i) = \mathrm{Log}\left(g_1^{e_1} \cdots g_l^{e_l}\right).$$

*Then $\bar{\alpha}_i$ is a polynomial in $e_1, \ldots, e_l$ for $i = 1, \ldots, l$.*

(ii) *Define $l$ functions $\bar{e}_1, \ldots, \bar{e}_l$ in $l$ rational variables $\alpha_1, \ldots, \alpha_l$ such that*

$$g_1^{\bar{e}_1} \cdots g_l^{\bar{e}_l} = \mathrm{Exp}\left(\sum_{i=1}^{l} \alpha_i \mathrm{Log}(g_i)\right).$$

*Then $\bar{e}_i$ is a polynomial in $\alpha_1, \ldots, \alpha_l$ for $i = 1, \ldots, l$.*

**Proof.** (i) Let $x = \sum_{i=1}^{l} r_i \mathrm{Log}(g_i)$, $y = \sum_{i=1}^{l} s_i \mathrm{Log}(g_i) \in \mathcal{L}(G)$ and $x * y = \sum_{i=1}^{l} t_i \mathrm{Log}(g_i)$. By the properties of the Baker–Campbell–Hausdorff formula, $t_i$ is a polynomial in $r_1, \ldots, r_l, s_1, \ldots, s_l$. For $g = g_1^{e_1} \cdots g_l^{e_l}$ we have that $\mathrm{Log}(g) = (e_1 \mathrm{Log}(g_1)) * \cdots * (e_l \mathrm{Log}(g_l))$. Therefore $\bar{\alpha}_i$ is a polynomial in $e_1, \ldots, e_l$.

(ii) Let $x = \sum_{i=1}^{l} \alpha_i \mathrm{Log}(g_i) \in \mathcal{L}(G)$. If $l = 1$, then $\bar{e}_1(\alpha_1) = \alpha_1$ and thus $\bar{e}_1$ is a polynomial. Now assume that $l > 1$. In Section 4.2 we saw that $(\bar{e}_1, \ldots, \bar{e}_l) = (\alpha_1, f_2, \ldots, f_l)$ where $g_2^{f_2} \cdots g_l^{f_l} = \mathrm{Exp}(y)$ with $y = (-\alpha_1 \mathrm{Log}(g_1)) * x$. Let $y = \sum_{i=2}^{l} \beta_i \mathrm{Log}(g_i)$. By the properties of the Baker–Campbell–Hausdorff formula $\beta_i$ is a polynomial in $\alpha_1, \ldots, \alpha_l$. Further by induction we can assume that $f_2, \ldots, f_l$ are polynomials in $\beta_2, \ldots, \beta_l$. Thus $\bar{e}_i$ is a polynomial in $\alpha_1, \ldots, \alpha_l$ for $i = 1, \ldots, l$. □

**Example 4.5.** Let $G = F_{2,2}$ be the group already studied in Examples 4.2 and 4.3. We have that

$$\mathrm{Log}\left(g_1^{e_1} g_2^{e_2} g_3^{e_3}\right) = \left(e_1 \mathrm{Log}(g_1)\right) * \left(e_2 \mathrm{Log}(g_2)\right) * \left(e_3 \mathrm{Log}(g_3)\right)$$

which is equal to $e_1 \mathrm{Log}(g_1) + e_2 \mathrm{Log}(g_2) + e_3 \mathrm{Log}(g_3) + \frac{1}{2}[e_1 \mathrm{Log}(g_1), e_2 \mathrm{Log}(g_2)]$. Therefore we have $\bar{\alpha}_1 = e_1$, $\bar{\alpha}_2 = e_2$ and $\bar{\alpha}_3 = -\frac{1}{2} e_1 e_2 + e_3$.

The proof of Lemma 4.4 is constructive and can be used to compute the polynomials $\bar{\alpha}_i$ and $\bar{e}_j$ if the structure constant table of the Lie algebra $\mathcal{L}(G)$ is known. See Section 4.4 for comments on the implementation and runtimes.

The functions $\bar{\alpha}_i$, $\bar{e}_j$ can be applied for the computation of logarithms and exponentials. In Section 4.4 we will see that this is yields a considerable speed up in comparison with the methods described in Sections 4.1 and 4.2.

## 4.4. Runtimes and comparison

The approaches described in Sections 4.1–4.3 to realizing the Mal'cev correspondence have been implemented in GAP [23] as a part of the package Guarana [2]. In this section we make comments on their implementation, indicate runtimes and compare them.

For the method of Section 4.1, we used the algorithm and implementation of Nickel [19] to compute the faithful matrix representations of the given $\mathcal{T}$-group $G$. It is much more efficient than previous methods [5,15].

For the method of Section 4.2, we use a *weight function* that can be associated to every Mal'cev basis $\mathcal{G} = (g_1, \dots, g_l)$; this is a function $w : \mathcal{G} \to \mathbb{N}\backslash\{0\}$ such that for all $g_k$ showing up in the normal form of $[g_i, g_j]$ we have that $w(g_k) \geqslant w(g_i) + w(g_j)$. If $\bar{w} = \max w(\mathcal{G})$ then $[g_i, g_j] = 1$ if $w(g_i) + w(g_j) > \bar{w}$; more general a group commutator in $g_i$ and $g_j$ with $\alpha$ occurrences of $g_i$ and $\beta$ occurrences of $g_j$ is equal to 1 if $\alpha w(g_i) + \beta w(g_j) > \bar{w}$. The equivalent fact holds in the Lie algebra. A Lie commutator in $\operatorname{Log}(g_i)$ and $\operatorname{Log}(g_j)$ with $\alpha$ occurrences of $\operatorname{Log}(g_i)$ and $\beta$ occurrences of $\operatorname{Log}(g_j)$ is equal to zero if $\alpha w(g_i) + \beta w(g_j) > \bar{w}$. This can be used to reduce the number of commutators which have to be evaluated during the computation of $\operatorname{Log}(g_i) * \operatorname{Log}(g_j)$.

For the method of Section 4.3 for computing Log and Exp, we use the structure constant table of the Lie algebra $\mathcal{L}(G)$ as computed by the method of Section 4.2. Further the terms of the BCH-formula are determined using [20, §8] and the Dynkin bracket operator defined in [25, Chapter 2].

We use the following two classes of examples of polycyclically presented $\mathcal{T}$-groups to test our implementations.

1. Let $\mathbb{Q}(\theta)$ be an algebraic extension of $\mathbb{Q}$ and $\mathcal{O}$ its maximal order. Then we denote by $\operatorname{Tr}_1(\mathcal{O})$ the group of upper-unitriangular matrices in $\mathrm{GL}_n(\mathcal{O})$. In a similar way to $\operatorname{Tr}_1(n, \mathbb{Q})$, we can compute a constructive polycyclic sequence for $\operatorname{Tr}_1(n, \mathcal{O})$, which then yields a polycyclic presentation for $\operatorname{Tr}_1(n, \mathcal{O})$. We use the irreducible polynomials $p_1(x) = x^2 - 3$ and $p_2(x) = x^3 - x^2 + 4$ for our examples. By $\mathcal{O}_i$ we denote the maximal order of $\mathbb{Q}(\theta_i)$ where $\theta_i$ is a zero of the polynomial $p_i$.

2. Let $F_n$ be the free group on $n$ generators $f_1, \dots, f_n$. Then $F_{n,c} = F_n/\gamma_{c+1}(F_n)$, where $\gamma_i$ denotes $i$th term of the lower central series, is the free nilpotent of class $c$ group on $n$ generators. It is a $\mathcal{T}$-group and we use the nilpotent quotient algorithm in the GAP package NQ [18] to compute a polycyclic presentation for it.

In Table 1 we indicate the time that is needed to set up the Mal'cev correspondence for several examples of $\mathcal{T}$-groups. Further the time that is needed to compute the polynomials $\bar{\alpha}_i, \bar{e}_j$, defined in Section 4.3, are displayed.

All computations where carried out in GAP Version 4.4.7 on a Pentium 4 with 3 gigahertz.

The runtimes displayed in Table 1 show that setting up the Mal'cev correspondence with the help of the BCH-formula, as described in Section 4.2, is more efficient than using matrix representations, as described in Section 4.1. For the tested examples the BCH-method is usually 100 to 1000 times faster than the matrix method.

Our experiments also show that the average time needed for computing Log and Exp using the method from Section 4.2 is faster than using the method from Section 4.1. Further the symbolic approach of Section 4.3 yields an additional speed up. For example for the group $F_{2,8}$ for random elements of range 1024 (i.e. elements of the form $g = g_1^{e_1} \cdots g_k^{e_k}$, where $e_i$ is a randomly chosen integer in $[-1024, \dots, 1024]$) computing Log costs only 10 milliseconds (symbolic) instead of 106 milliseconds (BCH) or 1979 milliseconds (Matrix approach). However also a considerable

Table 1
Setup of the Mal'cev correspondence and symbolic Log and Exp

| Group | Hl | Class | Matrix | BCH | Pols |
|---|---|---|---|---|---|
| $F_{2,2}$ | 3 | 2 | 16 | 4 | 8 |
| $F_{2,3}$ | 5 | 3 | 36 | 6 | 20 |
| $F_{2,4}$ | 8 | 4 | 80 | 8 | 36 |
| $F_{2,5}$ | 14 | 5 | 380 | 20 | 120 |
| $F_{2,6}$ | 23 | 6 | 1756 | 44 | 380 |
| $F_{2,7}$ | 41 | 7 | 14 825 | 196 | 1684 |
| $F_{2,8}$ | 71 | 8 | 154 000 | 776 | 6536 |
| $F_{3,2}$ | 6 | 2 | 32 | 4 | 20 |
| $F_{3,3}$ | 14 | 3 | 292 | 20 | 68 |
| $F_{3,4}$ | 32 | 4 | 3256 | 80 | 256 |
| $F_{3,5}$ | 80 | 5 | 97 239 | 820 | 1940 |
| $F_{3,6}$ | 196 | 6 | 3 504 971 | 8681 | 14 833 |
| $G(\mathrm{Tr}_1(2,\mathcal{O}_1))$ | 2 | 1 | 8 | 1 | 1 |
| $G(\mathrm{Tr}_1(3,\mathcal{O}_1))$ | 6 | 2 | 48 | 4 | 12 |
| $G(\mathrm{Tr}_1(4,\mathcal{O}_1))$ | 12 | 3 | 180 | 16 | 48 |
| $G(\mathrm{Tr}_1(5,\mathcal{O}_1))$ | 20 | 4 | 1048 | 68 | 108 |
| $G(\mathrm{Tr}_1(6,\mathcal{O}_1))$ | 30 | 5 | 5784 | 232 | 324 |
| $G(\mathrm{Tr}_1(7,\mathcal{O}_1))$ | 42 | 6 | 47 116 | 772 | 880 |
| $G(\mathrm{Tr}_1(8,\mathcal{O}_1))$ | 56 | 7 | 393 325 | 1988 | 2813 |
| $G(\mathrm{Tr}_1(2,\mathcal{O}_2))$ | 3 | 1 | 12 | 4 | 4 |
| $G(\mathrm{Tr}_1(3,\mathcal{O}_2))$ | 9 | 2 | 64 | 8 | 32 |
| $G(\mathrm{Tr}_1(4,\mathcal{O}_2))$ | 18 | 3 | 631 | 48 | 132 |
| $G(\mathrm{Tr}_1(5,\mathcal{O}_2))$ | 30 | 4 | 4969 | 188 | 432 |
| $G(\mathrm{Tr}_1(6,\mathcal{O}_2))$ | 45 | 5 | 32 630 | 664 | 2184 |
| $G(\mathrm{Tr}_1(7,\mathcal{O}_2))$ | 63 | 6 | 363 484 | 2069 | 11 972 |

The second and third column indicate the Hirsch Length and the class of the given example group. In the fourth, respectively fifth, column we display the time in milliseconds that is needed to set up the Mal'cev correspondence via the matrix approach (see Section 4.1), respectively the BCH approach (see Section 4.2). In the sixth column we see the time in milliseconds that is needed to compute the polynomials describing Log and Exp (see Section 4.3).

amount of time is needed to compute the polynomials used for the symbolic method. In the case of the group $F_{3,6}$ our implementation needs 15 seconds for the computation of these polynomials.

## 5. Collection in polycyclic groups

### 5.1. Classical collection

Let $G$ be a polycyclic group given by a polycyclic presentation with respect to a polycyclic sequence $\mathcal{G} = (g_1, \ldots, g_n)$. Let $w = w(g_1, \ldots, g_n)$ be a word in $g_1, \ldots, g_n$. A method for computing the normal form of $w$ with respect to $\mathcal{G}$ is called a *collection algorithm*. The performance of algorithms for computations in polycyclically presented groups depends very considerably on the ability to do collection efficiently. Typically the word $w$ is the product of two elements given in normal form, i.e. $w = \mathrm{nf}(g)\,\mathrm{nf}(h)$ for some $g, h \in G$.

Several strategies for collection in polycyclic groups have been studied intensively, see for example [8,13,24]. The current state of the art is "collection from the left." It consists of replacing repeatedly the leftmost uncollected subterm of the form $g_i^{r_i}$ or $g_i g_j^{\pm 1}$ where $j < i$ by a collected subterm according to the relations of the polycyclic presentations as displayed in Section 2. To our knowledge the fastest current implementation of collection from the left is part of MAGMA [4]. A description of this implementation can be found in [8].

The complexity of collection from the left is known to be exponential in the number $n$ of generators [13]. For collection with respect to certain "nice" polycyclic sequences, much better methods are known. For $\mathcal{T}$-groups, i.e. finitely generated torsion-free nilpotent groups, we have the following result due to Hall [9].

**Theorem 5.1.** *Let $G$ be a $\mathcal{T}$-group with Mal'cev basis $(g_1, \dots, g_l)$. Define the functions $\zeta_1, \dots, \zeta_l$ in $2l$ integer variables $x_1, \dots, x_l, y_1, \dots, y_l$ such that*

$$g_1^{x_1} \cdots g_l^{x_l} g_1^{y_1} \cdots g_l^{y_l} = g_1^{\zeta_1} \cdots g_l^{\zeta_l}$$

*and functions $\omega_1, \dots, \omega_l$ in $l+1$ integer variable $x_1, \dots, x_l, k$ such that*

$$\left(g_1^{x_1} \cdots g_l^{x_l}\right)^k = g_1^{\omega_1} \cdots g_l^{\omega_l}.$$

*Then the functions $\zeta_i$ and $\omega_j$ are rational polynomials.*

**Proof.** Recall that we denote $g_1^{\zeta_1} \cdots g_l^{\zeta_l}$ by $g^\zeta$. By the identities for Log in Section 3 we have that $g^\zeta = \mathrm{Exp}(\mathrm{Log}(g^x) * \mathrm{Log}(g^y))$ and $g^\omega = \mathrm{Exp}(k\mathrm{Log}(g_1^{x_1} \cdots g_l^{x_l}))$. By Lemma 4.4 and the fact that $x * y$ has only finitely many non-zero terms for $x, y \in \mathcal{L}(N)$, we deduce that $\zeta_i$ and $\omega_j$ are rational polynomials. $\square$

This result can be used for computational applications. In the 90s Leedham-Green and Soicher developed the algorithm "Deep Thought" [14], which computes these polynomials and uses them for collection in $\mathcal{T}$-groups. An implementation of Deep Thought by Merkwitz [17] is part of the GAP system. Deep Thought yields a big speed up compared to collection from the left for the multiplication of two random elements of a $\mathcal{T}$-group.

## *5.2. Collection using the Mal'cev correspondence*

Let $G$ be an infinite polycyclic group. In this section we show that, with respect to a carefully chosen polycyclic sequence $\mathcal{G}$ of $G$, collection in $G$ can be reduced to the following 3 subtasks:

(1) Collection and powering in a $\mathcal{T}$-group $N \lhd G$.
(2) Computations with powers of automorphisms and consecutive powers of automorphisms of a $\mathcal{T}$-group $N \lhd G$.
(3) Computations with coset representatives of a subgroup of finite index of $G$.

As discussed in Section 5.1, (1) is a well understood and we can apply standard methods such as Deep Thought to it. For (2) we will use the Mal'cev correspondence as explained in Sections 5.3 and 5.4. For (3) we use collection from the left.

The usage of the Mal'cev correspondence for collection in polycyclic groups is motivated by a paper of du Sautoy [6]. He uses the Mal'cev correspondence to investigate the nature of functions that describe the collection process in splittable polycyclic groups.

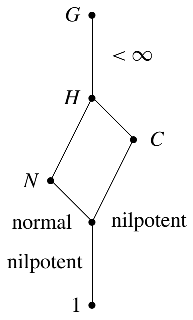

Fig. 1. Let $G$ be an infinite polycyclic group. It is well known that there exist a normal $\mathcal{T}$-group $N$ and a $\mathcal{T}$-group $C$ such that $H = CN$ is normal of finite index in $G$ and $H/N$ is free abelian of finite rank. In Section 5.2 we describe an effective collection method with respect to a polycyclic sequence $\mathcal{G}$ going through the normal series $1 \leqslant N \leqslant H \leqslant G$.

### *5.2.1. Nilpotent almost supplements*

We recall some facts about the structure of infinite polycyclic groups. It is well known that, if $G$ is a polycyclic group, then it has a normal series $G \geqslant K \geqslant N \geqslant 1$, where $N$ is a $\mathcal{T}$-group, $K/N$ is free-abelian and $G/K$ is finite. Further the next theorem due to Newell shows that, up to a finite index, the group $G$ is the product of two $\mathcal{T}$-groups. For a proof see [21, Chapter 3].

**Theorem 5.2.** *Let $K$ be a polycyclic group and $N$ a $\mathcal{T}$-group which is normal in $K$ and has the property that $K/N$ is nilpotent. Then there exists a $\mathcal{T}$-group $C$ such that $CN$ is of finite index in $K$.*

The group $C$ from Theorem 5.2 is said to be a *nilpotent almost-supplement* for $N$ in $K$. 'Almost' because $C$ and $N$ generate a subgroup of finite index, and 'supplement' because $C$ may intersect $N$ non-trivially. Since $[G : K] < \infty$ the group $C$ is also a nilpotent almost-supplement for $N$ in $G$. This structure of polycyclic groups can also be explored algorithmically. In [7, Chapter 9] Eick describes a practical algorithm to compute a nilpotent-by-abelian-by-finite series $G \geqslant K \geqslant N \geqslant 1$ and methods to determine a nilpotent almost-supplement $C$ for $N$ in $G$. Since $[G : CN] < \infty$ we can assume, by passing to the core of $CN$, that $CN$ is normal in $G$. See Fig. 1 for an illustration of Theorem 5.2.

### *5.2.2. Choosing the polycyclic sequence $\mathcal{G}$*

Now we explain how to chose the polycyclic sequence $\mathcal{G}$ mentioned at the beginning of Section 5.2. Let $\mathcal{N} = (n_1, \ldots, n_l)$ be a Mal'cev basis of $N$ and let $(c_1N, \ldots, c_kN)$ be a basis for the free abelian group $CN/N$. Then $\mathcal{H} = (c_1, \ldots, c_k, n_1, \ldots, n_l)$ is a polycyclic sequence for $H = CN$. Further there exists $f_1, \ldots, f_j \in G$ such that $(f_1H, \ldots, f_jH)$ is a polycyclic sequence for $G/H$. Now we set

$$\mathcal{G} = (f_1, \ldots, f_j, c_1, \ldots, c_k, n_1, \ldots, n_l)$$

which is a polycyclic sequence for $G$.

**Lemma 5.3.** *The list $(c_1, \ldots, c_k)$ can be extended to a Mal'cev basis*

$$\mathcal{C} = (c_1, \ldots, c_k, c_{k+1}, \ldots, c_{k+m})$$

*of the $\mathcal{T}$-group $C$.*

**Proof.** The upper central series of $C \cap N$ has torsion-free factors and is invariant under the action of $\langle c_1, \dots, c_k \rangle$. This series can be refined to a central series with torsion-free factors which are centralized by $\langle c_1, \dots, c_k \rangle$. To see this, let $M$ be one of the torsion-free factors. Denote by $A$ the centralizer of $\langle c_1, \dots, c_k \rangle$ in $M$. Then $A$ is non-trivial, since $C$ acts nilpotently on $M$, and furthermore $M/A$ is torsion-free, since for $m \in M, z \in \mathbb{N}$, the equality $zm = (zm)^{c_i} = z(m^{c_i})$ implies $m^{c_i} = m$. Thus, by induction on the dimension of $M$, we get a strictly ascending series of submodules of $M$ with torsion-free factors, which are by construction centralized by $\langle c_1, \dots, c_k \rangle$. □

*5.2.3. Mal'cev collection in $H = CN$*

Now we show that in $H$ collection with respect to $\mathcal{H}$ can be reduced to the subtasks (1) and (2). Denote by $c^x n^{\bar{x}}$ the element in $H$ given by the exponent vector $(x_1, \dots, x_k, \bar{x}_1, \dots, \bar{x}_l)$ with respect to $\mathcal{H}$. For two elements $c^x n^{\bar{x}}, c^y n^{\bar{y}} \in H$ we have

$$c^x n^{\bar{x}} c^y n^{\bar{y}} = c^x c^y \left(n^{\bar{x}}\right)^{(c^y)} n^{\bar{y}}.$$

Since $H/N$ is free abelian, the normal form of $c^x c^y$ with respect to $\mathcal{C}$ is of the form $c^{x+y} c_{k+1}^{z_{k+1}} \cdots c_{k+m}^{z_{k+m}}$. The computation of the tail $t = c_{k+1}^{z_{k+1}} \cdots c_{k+m}^{z_{k+m}}$ is a computation entirely in the $\mathcal{T}$-group $C$ and therefore a part of subtask (1).

We can also compute the normal form of the tail $t \in C \cap N$ with respect to $\mathcal{N}$ as part of (1). For this purpose we compute the normal forms of $c_{k+1}, \dots, c_{k+m}$ with respect to $\mathcal{N}$ as part of the setup. Then computing the normal form of $t$ with respect $\mathcal{N}$ reduces to $m$ powering operations and $m - 1$ multiplications in $N$.

The efficient computation of the normal form of

$$\left(n^{\bar{x}}\right)^{(c^y)} = \left(n^{\bar{x}}\right)^{(c_1^{y_1} \cdots c_k^{y_k})}$$

is the crucial step of our method. It is a computation with automorphisms of $N$ and therefore part of (2). Finally, the multiplication of $\mathrm{nf}(t), \mathrm{nf}((n^{\bar{x}})^{(c^y)})$ and $n^{\bar{y}}$ in $N$ can be done again as a part of (1).

*5.2.4. Inversion in $H = CN$*

Let $c^x n^{\bar{x}} \in H$ as in Section 5.2.3. The computation of the normal form of $g = (c^x n^{\bar{x}})^{-1}$ can be done as follows. Denoting $c = c^x$ and $n = n^{\bar{x}}$ we have that $g = c^{-1} (n^{-1})^{(c^{-1})}$. Inverting in $C$ and $N$ is part of (1), and thus the normal form of $c^{-1}$ with respect to $\mathcal{C}$ and the normal form of $n^{-1}$ with respect to $\mathcal{N}$ can be determined efficiently. Similarly to Section 5.2.3, we transform the normal form of $c^{-1}$ with respect to $\mathcal{C}$ to an element of the form $c^y n^{\bar{y}}$. The remaining computation of the normal form of $n^{\bar{y}} (n^{-1})^{(c^{-1})}$ can be done as in Section 5.2.3 and therefore be reduced to (1), (2).

*5.2.5. Powering in $H = CN$*

Let $c^x n^{\bar{x}} \in H$ as in Section 5.2.3. We describe a method to compute the normal form of $(c^x n^{\bar{x}})^q$ where $q \in \mathbb{Z}$.

By inverting $c^x n^{\tilde{x}}$ with the method of Section 5.2.4 if necessary, we can assume that $q \geqslant 0$. If we denote $c = c^x$ and $n = n^{\tilde{x}}$ then we have

$$(cn)^q = c^q \underbrace{n^{(c^{q-1})} \cdots n^c n}_{t}.$$

The group $H/N$ is free abelian. Thus $c^q = (c^x)^q = c_1^{qx_1} \cdots c_k^{qx_k} c_{k+1}^{z_{k+1}} \cdots c_{k+m}^{z_{k+m}}$ for some $z_{k+1}, \dots, z_{k+m} \in \mathbb{Z}$. The computation of the tail $s = c_{k+1}^{z_{k+1}} \cdots c_{k+m}^{z_{k+m}}$ is a powering computation in the $\mathcal{T}$-group $C$ and therefore a part of subtask (1). We can also compute the normal form of the tail $s \in C \cap N$ with respect to $\mathcal{N}$ as part of (1), as described in Section 5.2.3. The computation of the normal form of $t$ is a subtask of (2). Finally, the computation of the normal form of $st$ is again a part of (1).

### 5.2.6. *Mal'cev collection in G*

Now we describe our collection method with respect to $\mathcal{G}$. Denote by $f^x c^{\bar{x}} n^{\tilde{x}}$ the element in $G$ given by the exponent vector $(x_1, \dots, x_j, \bar{x}_1, \dots, \bar{x}_k, \tilde{x}_1, \dots, \tilde{x}_l)$ with respect to $\mathcal{G}$. For two elements $f^x c^{\bar{x}} n^{\tilde{x}}, f^y c^{\bar{y}} n^{\tilde{y}} \in G$ we have

$$f^x c^{\bar{x}} n^{\tilde{x}} f^y c^{\bar{y}} n^{\tilde{y}} = \underbrace{f^x f^y}_{f^r c^{\bar{r}} n^{\tilde{r}}} \underbrace{\left(c^{\bar{x}}\right)^{(f^y)}}_{c^{\bar{s}} n^{\tilde{s}}} \underbrace{c^{\bar{y}} \left(n^{\tilde{x}}\right)^{(f^y c^{\bar{y}})} n^{\tilde{y}}}_{c^{\bar{t}} n^{\tilde{t}}},$$

where $f^r c^{\bar{r}} n^{\tilde{r}}$, $c^{\bar{s}} n^{\tilde{s}}$ and $c^{\bar{t}} n^{\tilde{t}}$ are the normal forms with respect to $\mathcal{G}$ of the corresponding expressions in the brackets above them.

- The computation of $c^{\bar{t}} n^{\tilde{t}}$ can be reduced to the subtasks (1), (2) as explained in Section 5.2.3.
- For the computation of $c^{\bar{s}} n^{\tilde{s}}$ we use that

$$\left(c^{\bar{x}}\right)^{(f^y)} = \left(c_1^{(f^y)}\right)^{\bar{x}_1} \cdots \left(c_k^{(f^y)}\right)^{\bar{x}_k}.$$

  The normal form of $c_i^{(f^y)} \in H$ can be precomputed for $i = 1, \dots, k$ and every $f^y H \in G/H$, and therefore can be assumed to be given. Then $\mathrm{nf}((c_i^{(f^y)})^{\bar{x}_i})$ can be computed using the methods for powering in $H$ of Section 5.2.5. The remaining computation of the normal form of $\mathrm{nf}((c_1^{(f^y)})^{\bar{x}_1}) \cdots \mathrm{nf}((c_k^{(f^y)})^{\bar{x}_k})$ can be done by again using the methods for $H$.
- The normal form $f^r c^{\bar{r}} n^{\tilde{s}}$ of $f^x f^y$ can be precomputed for all $f^x H, f^y H \in G/H$ and therefore assumed to be given.
- Finally, the computation of the normal form of $c^{\bar{r}} n^{\tilde{r}} c^{\bar{s}} n^{\tilde{s}} c^{\bar{t}} n^{\tilde{t}}$ can be done with the method of Section 5.2.3.

### 5.2.7. *Inversion in G*

Let $g = f^x c^{\bar{x}} n^{\tilde{x}}$ be an element in $G$ given in normal form with respect to $\mathcal{G}$. Then $g^{-1} = (c^{\bar{x}} n^{\tilde{x}})^{-1} (f^x)^{-1}$. Thus, by precomputing the normal forms of all elements $(f^x)^{-1}$, we can reduce the computation of the normal form of $g^{-1}$ to inversion in $CN$ and collection in $G$.

## *5.3. Computations with powers of automorphisms of $\mathcal{T}$-groups*

Let $N$ be a $\mathcal{T}$-group given by a polycyclic presentation with respect to a Mal'cev basis $\mathcal{N} = (n_1, \dots, n_l)$. Let $\varphi$ be an automorphism of $N$, given by the list $(n_1^{\varphi}, \dots, n_l^{\varphi})$. In this section we describe an effective method to compute the normal form of $n^{(\varphi^q)}$, where $n \in N$ and $q \in \mathbb{Z}$.

As explained in Section 3, let $\mathcal{L}(N)$ be the Lie algebra corresponding to the radicable hull of $N$. Then $\{\mathrm{Log}(n_1), \dots, \mathrm{Log}(n_l)\}$ is a basis for $\mathcal{L}(N)$. We define a $l \times l$ matrix $\Phi$ by

$$\mathrm{Log}(n_i^{\varphi}) = \sum_{j=1}^{l} \Phi_{ij} \mathrm{Log}(n_i).$$

By Theorem 3.4 the matrix $\Phi$ is a representation of the Lie algebra isomorphism $\mathrm{Exp} \circ \varphi \circ \mathrm{Log}$, with respect to the basis $\{\mathrm{Log} n_1, \dots, \mathrm{Log} n_l\}$. This yields the following algorithm.

**ApplyPowerOfAutomorphism($n$, $\varphi$, $q$)**

```
1: determine $\gamma = $ **Logarithm($n$)**.
2: compute $\bar{\gamma} = \gamma \cdot \Phi^q$.
3: compute $g = $ **Exponential($\bar{\gamma}$)**.
4: return $g$.
```

For the realization of Steps 1 and 3, see Section 4. Note that for Step 2 repeated squaring can be used.

If we want to apply several powers of automorphisms $\varphi_1^{q_1} \varphi_2^{q_2} \cdots \varphi_k^{q_k}$, as this is the case in Section 5.2, we switch only once from $n$ to the corresponding element $\gamma$ in the Lie algebra, then multiply $\gamma$ with $\Phi_1^{q_1} \cdots \Phi_k^{q_k}$, where $\Phi_i$ is the matrix representation of the Lie algebra isomorphism corresponding to the group automorphism $\varphi_i$, and then switch back to the representation with respect to $(n_1, \dots, n_l)$.

## *5.4. Computations with consecutive powers of automorphisms of $\mathcal{T}$-groups*

Let $N$ be a $\mathcal{T}$-group and $\varphi \in \mathrm{Aut}(N)$ be given as in Section 5.3. We describe an effective method to compute the normal form of

$$\pi_{q+1} = n^{(\varphi^q)} n^{(\varphi^{q-1})} \cdots n^{\varphi} n$$

where $n \in N$ and $q \in \mathbb{N}$.

As in Section 5.3 we use the Lie algebra $\mathcal{L}(N)$ and the Lie algebra automorphism $\Phi \in \mathrm{Aut}(\mathcal{L}(N))$ corresponding to $\varphi$ for this purpose; if we denote $x = \log(n)$ then we are interested in computing the coefficients of the vector

$$\Pi_{q+1} = (x\Phi^q) * (x\Phi^{(q-1)}) * \cdots * (x\Phi) * x$$

because $\mathrm{Exp}(\Pi_{q+1}) = \pi_{q+1}$.

Our method is a variation of repeated squaring; we use a binary representation of $q$ and the identities

$$\Pi_{2p} = (\Pi_p \Phi^p) * \Pi_p,$$
$$\Pi_{2p+1} = (\Pi_{2p} \Phi) * x.$$

This reduces the computations of the coefficients of $\Pi_q$ to $\log(q)$ matrix and vector multiplications and $\log(q)$ $*$-operations.

## 5.5. *Implementation and runtimes*

The Mal'cev collection algorithm has been fully implemented in GAP and is part of the Guarana package [2]. In this section we make comments on our implementation and compare it with collection from the left. All computations have been carried out on a Pentium 4 machine with 3 gigahertz. Indications on memory usage will be given later at the appropriate places.

### 5.5.1. *Used example groups*

Throughout this section we use the following classes of example groups.

1. Let $\mathbb{Q}(\theta)$ be an algebraic extension of $\mathbb{Q}$ and $\mathcal{O}$ its maximal order. Let $\mathrm{Tr}_n(\mathcal{O})$ be the group of upper-triangular matrices in $\mathrm{GL}_n(\mathcal{O})$, $\mathrm{Tr}_1(n, \mathcal{O})$ the subgroup of matrices in $\mathrm{Tr}_n(\mathcal{O})$ with 1s on the diagonal and $D_n(\mathcal{O})$ the group of diagonal matrices in $\mathrm{GL}_n(\mathcal{O})$. Every polycyclic group has a subgroup of finite index which can be embedded in some $\mathrm{Tr}_n(\mathcal{O})$ [21, p. 132]. Therefore this class of groups is very suitable for testing our collection algorithm.

Let $U(\mathcal{O})$ be the group of units of $\mathcal{O}$. As a consequence of Dirichlet's Units Theorem, $U(\mathcal{O})$ is polycyclic and therefore also $D_n(\mathcal{O})$. Using the torsion unit and fundamental units of $U(\mathcal{O})$, it is straightforward to obtain a polycyclic presentation for $D_n(\mathcal{O})$.

As mentioned in Section 4.4 we can compute a polycyclic presentation for $\mathrm{Tr}_1(n, \mathcal{O})$. Since $\mathrm{Tr}_n(\mathcal{O}) = D_n(\mathcal{O}) \ltimes \mathrm{Tr}_1(n, \mathcal{O})$, it is straightforward to obtain a polycyclically presented group $G(\mathrm{Tr}_n(\mathcal{O}))$ being isomorphic to $\mathrm{Tr}_n(\mathcal{O})$.

We use the irreducible polynomials $p_1(x) = x^2 - 3$ and $p_2(x) = x^3 - x^2 + 4$ for our examples. By $\mathcal{O}_i$ we denote the maximal order of $\mathbb{Q}(\theta_i)$ where $\theta_i$ is a zero of the polynomial $p_i$.

2. Let $F_{n,c}$ be the free nilpotent of class $c$ group on $n$ generators. As explained in Section 4.4 we can compute a polycyclic presentation for $F_{n,c}$. An automorphism $\varphi$ of the free group $F_n$ naturally induces an automorphism $\bar{\varphi}$ of $F_{n,c}$.

We use the automorphism $\varphi_1$ of $F_2$ which maps $f_1$ to $f_2^{-1}$ and $f_2$ to $f_1 f_2^3$ and the automorphism $\varphi_2$ of $F_3$ mapping $f_1$ to $f_2^{-1}$, $f_2$ to $f_3^{-1}$ and $f_3$ to $f_2^{-3} f_1^{-1}$ for our examples.

Using an automorphism $\psi$ of $F_{n,c}$ we can construct a polycyclically presented group $G(\langle\psi\rangle \ltimes F_{n,c})$ which is isomorphic to $\langle\psi\rangle \ltimes F_{n,c}$.

### 5.5.2. *Runtimes setup*

In Table 2 we display the time that is needed for the complete setup of the Mal'cev collector. We assume that the input group is given by a polycyclic presentation with respect to a nice polycyclic sequence in the sense of Section 5.2.2 and that the subgroup $C$ is given by a polycyclic presentation with respect to a Mal'cev basis. The setup of the Mal'cev collector includes the setup of the Mal'cev correspondence for the normal subgroup $N$ as described in Section 4.2, the computation of the polynomials describing Log and Exp as described in 4.3, the computation of the multiplication table of $G/CN$, the computation of the Deep Thought collector for $C$ and $N$ and all other information that is needed to do Mal'cev collection. All computations have been carried out with 80 MB of memory for GAP.

Table 2
Setup of the Mal'cev collector

| Group | Hl | Setup time |
|---|---|---|
| $G(\langle\bar{\varphi_1}\rangle \ltimes F_{22})$ | 4 | 40 |
| $G(\langle\bar{\varphi_1}\rangle \ltimes F_{23})$ | 6 | 48 |
| $G(\langle\bar{\varphi_1}\rangle \ltimes F_{24})$ | 9 | 48 |
| $G(\langle\bar{\varphi_1}\rangle \ltimes F_{25})$ | 15 | 160 |
| $G(\langle\bar{\varphi_1}\rangle \ltimes F_{26})$ | 24 | 508 |
| $G(\langle\bar{\varphi_1}\rangle \ltimes F_{27})$ | 42 | 2232 |
| $G(\langle\bar{\varphi_1}\rangle \ltimes F_{28})$ | 72 | 8447 |
| $G(\langle\bar{\varphi_2}\rangle \ltimes F_{32})$ | 7 | 30 |
| $G(\langle\bar{\varphi_2}\rangle \ltimes F_{33})$ | 15 | 96 |
| $G(\langle\bar{\varphi_2}\rangle \ltimes F_{34})$ | 33 | 543 |
| $G(\langle\bar{\varphi_2}\rangle \ltimes F_{35})$ | 81 | 4201 |
| $G(\langle\bar{\varphi_2}\rangle \ltimes F_{36})$ | 197 | 41 376 |
| $G(\mathrm{Tr}_2(\mathcal{O}_1))$ | 4 | 16 |
| $G(\mathrm{Tr}_3(\mathcal{O}_1))$ | 9 | 48 |
| $G(\mathrm{Tr}_4(\mathcal{O}_1))$ | 16 | 148 |
| $G(\mathrm{Tr}_5(\mathcal{O}_1))$ | 25 | 428 |
| $G(\mathrm{Tr}_6(\mathcal{O}_1))$ | 36 | 1272 |
| $G(\mathrm{Tr}_7(\mathcal{O}_1))$ | 49 | 3864 |
| $G(\mathrm{Tr}_8(\mathcal{O}_1))$ | 64 | 12 757 |
| $G(\mathrm{Tr}_2(\mathcal{O}_2))$ | 5 | 28 |
| $G(\mathrm{Tr}_3(\mathcal{O}_2))$ | 12 | 76 |
| $G(\mathrm{Tr}_4(\mathcal{O}_2))$ | 22 | 292 |
| $G(\mathrm{Tr}_5(\mathcal{O}_2))$ | 35 | 1036 |
| $G(\mathrm{Tr}_6(\mathcal{O}_2))$ | 51 | 3949 |
| $G(\mathrm{Tr}_7(\mathcal{O}_2))$ | 70 | 17 197 |

In the second column we indicate the Hirsch length of the given example group. In the third column we specify the time in milliseconds that is needed for the complete setup of the Mal'cev collector.

### 5.5.3. *Mal'cev collection versus collection from the left*

In Table 3 we display the average runtime for the multiplication of two random elements in our example groups. The compared methods are collection from the left as implemented in MAGMA V2.12-14 and Mal'cev collection. For a group $G$ with polycyclic sequence $(g_1, \dots, g_k)$ we say that a random element $g \in G$ is of *range* $r \in \mathbb{N}$ if is of the form $g = g_1^{e_1} \cdots g_k^{e_k}$, where $e_i$ is a randomly chosen integer in $[-r, \dots, r]$. The Mal'cev collector uses the implementation of Deep Thought in GAP as the collection method in the $\mathcal{T}$-groups $C$ and $N$.

In Table 3 we see that Cftl is more efficient than Mal'cev collection for random elements of very small range such as 1. This is not surprising. For elements $g, h$ of small range Cftl needs to do very few replacements to yield the normal form of $gh$. We note that the runtime of Cftl can differ considerably because the size of the exponents of the normal form of $gh$ varies a lot for random elements $g, h$ of same range.

For random elements of bigger range Mal'cev collection dramatically outperforms Cftl. It is much faster and also less memory consuming. The multiplication of random elements of big range such as 1000 was not possible with Cftl with 1 GB of memory, while Mal'cev needed at most 85 MB of memory. See Fig. 2 for a graphical comparison of Mal'cev and Cftl for one example group.

Table 3
Runtimes for the multiplication of two random elements

| Group | Range = 1 | | Range = 10 | | Range = 100 | | Range = 1000 | |
|---|---|---|---|---|---|---|---|---|
| | Cftl | Mal'cev | Cftl | Mal'cev | Cftl | Mal'cev | Cftl | Mal'cev |
| $G(\langle\bar{\varphi}_1\rangle \ltimes F_{22})$ | 1 | 1 | 3 | 1 | 2481 | 1 | $*$ | 1 |
| $G(\langle\bar{\varphi}_1\rangle \ltimes F_{23})$ | 1 | 1 | 27 | 1 | $*$ | 2 | $*$ | 3 |
| $G(\langle\bar{\varphi}_1\rangle \ltimes F_{24})$ | 1 | 2 | 108 | 3 | $*$ | 4 | $*$ | 11 |
| $G(\langle\bar{\varphi}_1\rangle \ltimes F_{25})$ | 1 | 5 | 5627 | 7 | $*$ | 14 | $*$ | 151 |
| $G(\langle\bar{\varphi}_1\rangle \ltimes F_{26})$ | 1 | 11 | 21 276 | 19 | $*$ | 57 | $*$ | 810 |
| $G(\langle\bar{\varphi}_1\rangle \ltimes F_{27})$ | 2 | 52 | 485 532 | 120 | $*$ | 501 | $*$ | 9269 |
| $G(\langle\bar{\varphi}_1\rangle \ltimes F_{28})$ | 17 | 150 | $*$ | 522 | $*$ | 2442 | $*$ | 58 497 |
| $G(\langle\bar{\varphi}_2\rangle \ltimes F_{32})$ | 1 | 1 | 7 | 1 | $*$ | 2 | $*$ | 3 |
| $G(\langle\bar{\varphi}_2\rangle \ltimes F_{33})$ | 1 | 3 | 252 | 4 | $*$ | 7 | $*$ | 40 |
| $G(\langle\bar{\varphi}_2\rangle \ltimes F_{34})$ | 1 | 9 | 3160 | 13 | $*$ | 63 | $*$ | 568 |
| $G(\langle\bar{\varphi}_2\rangle \ltimes F_{35})$ | 1 | 62 | 325 312 | 158 | $*$ | 1185 | $*$ | 17473 |
| $G(\langle\bar{\varphi}_2\rangle \ltimes F_{36})$ | 64 | 434 | $*$ | 2268 | $*$ | 22 377 | $*$ | 310 518 |
| $G(\mathrm{Tr}_2(\mathcal{O}_1))$ | 1 | 1 | 1 | 2 | 4 | 3 | 67 | 3 |
| $G(\mathrm{Tr}_3(\mathcal{O}_1))$ | 1 | 5 | 2 | 7 | 20 | 9 | 308 | 11 |
| $G(\mathrm{Tr}_4(\mathcal{O}_1))$ | 1 | 9 | 19 | 14 | 978 | 16 | $*$ | 28 |
| $G(\mathrm{Tr}_5(\mathcal{O}_1))$ | 1 | 20 | 140 | 28 | 9068 | 35 | $*$ | 81 |
| $G(\mathrm{Tr}_6(\mathcal{O}_1))$ | 5 | 56 | 549 | 83 | 27 510 | 112 | $*$ | 391 |
| $G(\mathrm{Tr}_7(\mathcal{O}_1))$ | 28 | 127 | 1899 | 190 | 150 030 | 276 | $*$ | 1043 |
| $G(\mathrm{Tr}_8(\mathcal{O}_1))$ | 80 | 496 | 5884 | 785 | $*$ | 1381 | $*$ | 5450 |
| $G(\mathrm{Tr}_2(\mathcal{O}_2))$ | 1 | 2 | 1 | 3 | 9 | 3 | 187 | 4 |
| $G(\mathrm{Tr}_3(\mathcal{O}_2))$ | 1 | 5 | 3 | 8 | 47 | 10 | 732 | 15 |
| $G(\mathrm{Tr}_4(\mathcal{O}_2))$ | 1 | 15 | 85 | 23 | 6705 | 28 | $*$ | 72 |
| $G(\mathrm{Tr}_5(\mathcal{O}_2))$ | 4 | 43 | 829 | 59 | 34 520 | 77 | $*$ | 216 |
| $G(\mathrm{Tr}_6(\mathcal{O}_2))$ | 25 | 157 | 3233 | 225 | $*$ | 318 | $*$ | 1238 |
| $G(\mathrm{Tr}_7(\mathcal{O}_2))$ | 114 | 501 | 12 118 | 700 | $*$ | 1008 | $*$ | 3870 |

This table specifies the average runtime in milliseconds of 1000 computations of the normal form of $gh$ where $g, h$ are randomly chosen group elements of range $r$, i.e. elements of the form $g = g_1^{e_1} \cdots g_k^{e_k}$, where $e_i$ is a randomly chosen integer in $[-r, \dots, r]$. The two compared methods are Cftl, i.e. collection from the left as implemented in MAGMA V2.12-14 and Mal'cev collection as implemented in the GAP package Guarana. The symbol '$*$' indicates that the computation of the average runtime was aborted because it needed more than 1 GB of memory.

Therefore it depends very much on the context which method should be applied. For computations where typically elements with sparse and small exponent vectors are multiplied, Cftl is preferable. For example for the computation of a polycyclic sequence for the derived subgroup it might better to choose Cftl, since we mainly compute normal forms of commutators in the original generators. Computations that involve slightly more complicated elements are better done with Mal'cev collection. For those elements Mal'cev is much faster even if we include the cost of the setup of Mal'cev collector. Further the multiplication of elements of big range is often not possible with Cftl since we run out of memory.

It might be a good idea to use a hybrid collector that combines both methods. By looking at the exponent vectors of the input elements this collector could make an estimate whether Cftl or Mal'cev should be chosen. Alternatively the hybrid collector could run Cftl and Mal'cev in parallel on the same input and return the result of the method that finished first.

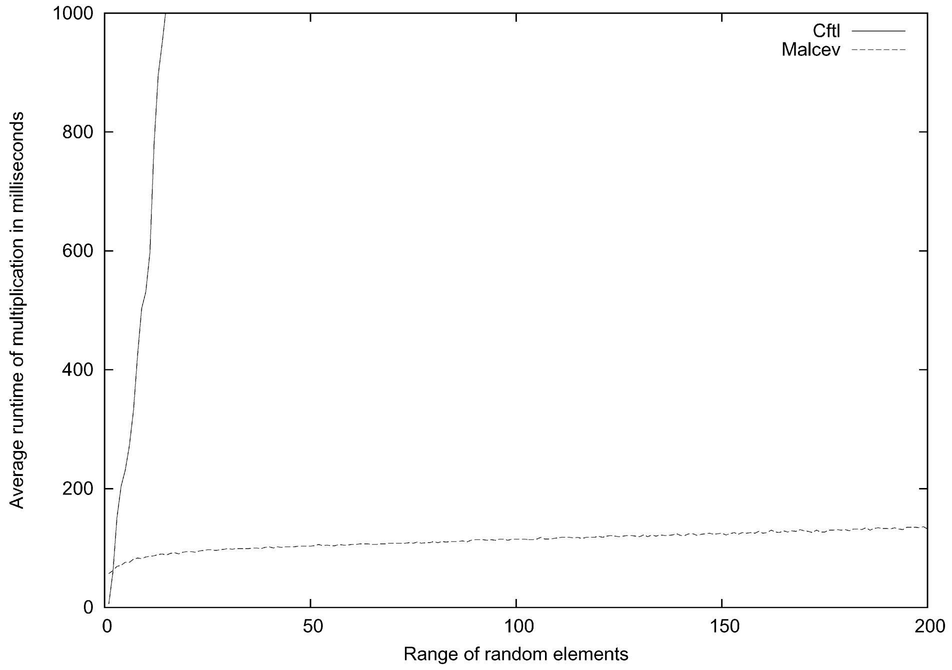

Fig. 2. For the group $G(\mathrm{Tr}_6(\mathcal{O}_1))$ the average runtime of the collection algorithm Mal'cev and Cftl as a function in the range of the multiplied random elements is displayed.

## *5.6. Open problems and future work*

We have seen in Section 5.5.3 that Mal'cev collection can be used to speed up collection in polycyclically groups considerably. However this collection methods only works if the group is given by a polycyclic presentation with respect to a nice polycyclic sequence in the sense of Section 5.2.2.

As mentioned in Section 5.2.1 the methods in [7, Chapter 9] can be used to compute such a nice polycyclic sequence for an arbitrary polycyclically presented group. Therefore it would be desirable to have a very efficient implementation of these methods. Since such a polycyclic sequence which goes through a nilpotent-by-abelian-by finite normal series of the group is a natural starting point for further investigations, it is desirable to compute it anyway.

Further it is likely that the Mal'cev correspondence can be used directly for a variety of algorithms dealing with infinite polycyclically presented groups.

## References

[1] B. Assmann, Algorithmic use of the Mal'cev correspondence, in: Groups St. Andrews 2005, in: London Math. Soc. Lecture Note Ser., vol. 339, Cambridge University Press, 2007, pp. 158–169.
[2] B. Assmann, Guarana—Applications of Lie methods in computational group theory, 2006, A GAP 4 package, see [23].
[3] G. Baumslag, Lecture Notes on Nilpotent Groups, Amer. Math. Soc., Providence, 1971.
[4] W. Bosma, J. Cannon, C. Playoust, The MAGMA algebra system I: The user language, J. Symbolic Comput. 24 (1997) 235–265.

[5] W. de Graaf, W. Nickel, Constructing faithful representations of finitely-generated torsion-free nilpotent groups, J. Symbolic Comput. 33 (2002) 31–41.
[6] M. du Sautoy, Polycyclic groups, analytic groups and algebraic groups, Proc. London Math. Soc. (3) 85 (2002) 62–92.
[7] B. Eick, Algorithms for polycyclic groups, Habilitationsschrift, Universität Kassel, 2001.
[8] V. Gebhardt, Efficient collection in infinite polycyclic groups, J. Symbolic Comput. 34 (3) (2002) 213–228.
[9] P. Hall, Nilpotent groups, in: The Collected Works of Philip Hall, Clarendon Press, Oxford, 1988, pp. 415–462. Notes of lectures given at the Canadian Mathematical Congress 1957 Summer Seminar.
[10] D.F. Holt, B. Eick, E.A. O'Brien, Handbook of Computational Group Theory, Discrete Math. Appl., CRC Press, 2005.
[11] M. Kargapolov, J. Merzljakov, Fundamentals of the Theory of Groups, Grad. Texts in Math., Springer, 1979.
[12] E. Khukhro, $p$-Automorphisms of Finite $p$-Groups, London Math. Soc. Lecture Note Ser., vol. 246, London Math. Soc., 1998.
[13] C.R. Leedham-Green, L.H. Soicher, Collection from the left and other strategies, J. Symbolic Comput. 9 (1990) 665–675.
[14] C.R. Leedham-Green, L.H. Soicher, Symbolic collection using Deep Thought, LMS J. Comput. Math. 1 (1998) 9–24.
[15] E.H. Lo, G. Ostheimer, A practical algorithm for finding matrix representations for polycyclic groups, J. Symbolic Comput. 28 (1999) 339–360.
[16] A.J. Mal'cev, On certain classes of infinite soluble groups, Mat. Sb. 28 (1951) 567–588.
[17] W. Merkwitz, Symbolische Multiplikation in nilpotenten Gruppen mit Deep Thought, Diplomarbeit, RWTH Aachen, 1997.
[18] W. Nickel, NQ, 1998, A refereed GAP 4 package, see [23].
[19] W. Nickel, Matrix representations for torsion-free nilpotent groups by Deep Thought, J. Algebra 300 (2006) 603–626.
[20] M. Reinsch, A simple expression for the terms of the Baker–Campbell–Hausdorff series, J. Math. Phys. 41 (4) (2000) 2434–2442.
[21] D. Segal, Polycyclic Groups, Cambridge Univ. Press, Cambridge, 1983.
[22] C.C. Sims, Computation with Finitely Presented Groups, Cambridge Univ. Press, Cambridge, 1994.
[23] The GAP Group, GAP—Groups, algorithms and programming, http://www.gap-system.org, 2006.
[24] M. Vaughan-Lee, Collection from the left, J. Symbolic Comput. 9 (1990) 725–733.
[25] M. Vaughan-Lee, The Restricted Burnside Problem, London Math. Soc. Monogr. (N.S.), vol. 5, Oxford Univ. Press, Oxford, 1990.