Computing nilpotent quotients in finitely presented Lie rings

Csaba Schneider

Discrete Mathematics and Theoretical Computer Science **1**, (1997), 1–16

Computing nilpotent quotients in finitely presented Lie rings†

Csaba Schneider

School of Mathematical Sciences, The Australian National University, ACT 0200 Canberra, Australia
E-Mail: C.Schneider@maths.anu.edu.au

A nilpotent quotient algorithm for finitely presented Lie rings over $\mathbb{Z}$ (LIENQ) is described. The paper studies the graded and non-graded cases separately. The algorithm computes the so-called nilpotent presentation for a finitely presented, nilpotent Lie ring. A nilpotent presentation consists of generators for the abelian group and the products expressed as linear combinations for pairs formed by generators. Using that presentation the word problem is decidable in L. Provided that the Lie ring L is graded, it is possible to determine the canonical presentation for a lower central factor of L. LIENQ's Complexity is studied and it is shown that optimising the presentation is NP-hard. Computational details are provided with examples, timing and some structure theorems obtained from computations. Implementation in C and GAP interface are available

Keywords: Lie rings, nilpotent Lie rings, finitely presented Lie rings, nilpotent presentation

1 Introduction

The nilpotent quotient algorithm for finitely presented Lie rings – LIENQ as we shall refer to it in the sequel – operates with Lie rings over $\mathbb{Z}$. Its essential goal is to compute an efficient and useful presentation for the nilpotent factors of a given Lie ring. It actually means to compute a presentation for its additive group and to compute the structure constants in order to determine the Lie ring structure. By efficient and useful we mean that several important pieces of information can be read off immediately (e.g.: nilpotency, nilpotency class etc). Furthermore, by this presentation the so-called *word problem* is decidable, although it is known to be undecidable in the general case. On word problem in groups and Lie algebras see [4].

Such algorithms have existed for several decades for groups and Lie rings as well. Recall, the most important commutator identities, such as

$$[x, y] = [y, x]^{-1} \tag{1}$$

$$[[x, y^{-1}], z]^y [[y, z^{-1}], x]^z [[z, x^{-1}], y]^x = 1 \tag{2}$$

$$[xy, z] = [x, z]^y [y, z] \tag{3}$$

$$[x, yz] = [x, z][x, y]^z \tag{4}$$

†The research presented in this paper was sponsored by the TEMPUS and HCM foundations of the EU.

reminds us of the defining axioms of Lie rings. In details, (2) is \similar" to the Jacobi identity while (3) and (4) can be interpreted as some \distributive" property of the group commutator. This analogy allows us to alter the known group algorithms for Lie rings.

The algorithm described here is a variation of those presented in [11] and [14], however it differs at several points from them.

There are three main sections in this paper. The first part contains the basic properties of the nilpotent presentation and describes an algorithm to compute it in the general case. The next part is devoted to graded Lie rings. We shall see that in the graded case we can simplify LIENQ and, using other techniques, it is possible to determine the isomorphism type of the underlying abelian group of such a Lie ring. The last part contains computational issues. We shall see how the implementation works and some examples with timing will display the scope of LIENQ in both graded and non-graded cases.

The basic properties of Lie rings can be found in any textbook on that topic. We refer to [12] for details. The first lemma is of fundamental importance to our goal, since it essentially shows that it is realistic to build a nilpotent quotient algorithm for finitely presented Lie rings.

Lemma 1.1 *In a finitely presented Lie ring each lower central factor is finitely presented.*

The lemma is a consequence of the argument presented in [5] on basic commutators.

2 The Nilpotent Presentation of a Lie Ring

Let L denote a Lie ring and L^i the ith term of its lower central series. Product in L is denoted by brackets – as it is usual – and we use the left-normed convention that is, $[a,b,c] = [[a,b],c]$. If L is finitely presented Lemma 1.1 implies that L^i/L^{i+1} is finitely presented for each i. Since at the present stage we are only interested in the nilpotent factor L/L^k, we think L is nilpotent of class $k-1$. In other words we think L/L^k is equal to L. In this case L has a so-called *nilpotent presentation* in accordance with the following definition.

Definition 2.1 *A nilpotent presentation is a tuple $(\mathcal{G}, S)$, where $\mathcal{G}$ is a finite set of generators, $\{a_1, \ldots, a_r\}$ say, and S is a finite set of relations of the form*

$$\gamma_i \cdot a_i = \alpha_{i,i+1} \cdot a_{i+1} + \cdots + \alpha_{i,r} \cdot a_r \quad \textit{for some } i, \qquad \textit{TR}$$

$$[a_j, a_i] = \beta_{j,i,j+1} \cdot a_{j+1} + \cdots + \beta_{j,i,r} \cdot a_r \quad \textit{for each } j > i, \qquad \textit{PR}$$

where the α, β, γ are integer coefficients, furthermore, $\gamma_i > 1$ where applicable.

Note that the relations of type TR are sometimes referred to as *torsion relations*, while those of PR are called *product relations*. A more detailed description and the proof for existence of such presentation can be found in [18], while it is easy to see that a nilpotent presentation always defines a nilpotent Lie ring. Observe that we demand the existence of PR relations for each pair (j,i), where $j > i$, but only some of those of type TR (even possibly none). Throughout this paper I denotes the set of indices, so that $i \in I$ if and only if a_i has TR relation.

One easily sees that the relations of type TR determine the underlying abelian group structure for L, while the relations of type PR define its ring structure. The β can be viewed as structure constants for the $\mathbb{Z}$-module defined by $\mathcal{G}$ and the TR relations.

For our purpose this concept will be too general, so we shall keep some restrictions. Define a weight function $\omega : \mathcal{G} \to \mathbb{N}$ as follows. Let ω be an increasing function in i, such that the following holds.

1. $\omega(a_1) = 1$,

2. if the nilpotent presentation contains a relation of the form

$$[a_j, a_i] = w_{ji},$$

where w_{ji} is a sum of integer multiples of generators from $\mathcal{G}$, then all generators $a_k \in \mathcal{G}$ such that $\omega(a_k) < \omega(a_j) + \omega(a_i)$ have coefficients zero in w_{ji}.

Another restriction, if $a_k \in \mathcal{G}$ and $\omega(a_k) > 1$, we require a_k to have a definition, i.e. a relation of the form

$$[a_j, a_i] = a_k \tag{5}$$

where $\omega(a_k) = \omega(a_j) + \omega(a_i)$ and $\omega(a_i) = 1$. It might easily happen that a nilpotent presentation contains more then one relation of the form (5) for some k, in this case we arbitrarily choose one for the definition of a_k.

If for a nilpotent presentation there exists a function ω which satisfies the conditions 1. and 2. then it is called a weighted nilpotent presentation with weight function ω.

Suppose that we are given a nilpotent presentation

$$\langle \mathcal{G} \mid \gamma_i \cdot a_i = w_i,\ i \in I, [a_j, a_i] = w_{ji},\ 1 \le i < j \le r \rangle. \tag{6}$$

Let L denote the Lie ring presented by (6) and A the anti-commutative, non-associative ring defined by (6). Clearly (6) does not care about Jacobi identities, so L does not automatically coincide with A.

It is important in the sequel to decide if $L \cong A$, in other words if A is a Lie ring. To do that, we arrange a collection process in A. Suppose that we are given a sum of products $\ell \in A$ expressed in terms of the elements from $\mathcal{G}$. The collection process consists of two main steps. In the first step we substitute the products by the right-hand side of the appropriate PR relations. In the second step we reduce those coefficients that are not less than the corresponding γ_i in (6).

After proceeding so, we get a reduced form for $\ell \in A$ subject to (6). That is,

$$\ell = \alpha_1 \cdot a_1 + \cdots + \alpha_r \cdot a_r \tag{7}$$

where $0 \le \alpha_i < \gamma_i$ whenever $i \in I$. This reduced form is unique in A, but the same element in L can be represented by a more reduced form subject to the nilpotent presentation. We do not want this phenomenon to happen at all, thus we define consistency of the nilpotent presentation.

Definition 2.2 *A weighted nilpotent presentation for a Lie ring L is said to be consistent if every element of L has a unique normal form.*

In Section 3.3 we shall see how to check if an arbitrary nilpotent presentation is consistent and develop a method to make it consistent if it is not. In general, it is an equivalent condition to that of Definition 2.2 that the element 0 uniquely has a normal form.

From the computational point of view it is important to see that not all the relations of type PR are necessary to determine the structure of L. Our first result is devoted to this observation.

Lemma 2.1 *Let L be a Lie ring given by the consistent weighted nilpotent presentation*

$$\langle \mathcal{G} \mid \gamma_i \cdot a_i = w_i,\ i \in I, [a_j, a_i] = w_{ji},\ 1 \leq i < j \leq r \rangle.$$

Then L is already determined by the presentation

$$\langle \mathcal{G} \mid \gamma_i \cdot a_i = w_i,\ i \in I,\ [a_j, a_i] = w_{ji},\ 1 \leq i < j \leq r,\ \omega(a_i) = 1 \rangle. \tag{8}$$

Proof. Essentially we want to prove that all PR relations can be expressed by those listed in (8). We proceed by an induction argument on $\omega(a_i)$. If $\omega(a_i) = 1$ then we are done. Suppose now, that $\omega(a_i) = k > 1$ and we have already expressed all relations of the form

$$[a_m, a_n] = w_{ji} \text{ where } \omega(a_n) < k.$$

Using the definition for a_i and the identities of a Lie ring one has: $[a_j, a_i] = [a_j, [a_k, a_l]] = -[a_k, a_l, a_j] = [a_l, a_j, a_k] + [a_j, a_k, a_l] = [a_j, a_k, a_l] - [a_j, a_l, a_k]$. And the latter ones are known since both a_k and a_l have weight smaller than k, hence by the induction hypothesis and linearity result we obtain an expression for $[a_j, a_i]$. Note that in the first equality we substituted a_i by its definition. □

3 Computing a Weighted Nilpotent Presentation

The aim of this section is to show how it is possible to compute a consistent weighted nilpotent presentation for a nilpotent factor of a a given finitely presented Lie ring L and determine an epimorphism from L onto that nilpotent factor.

The basic ideas – besides being simple – are similar to those used in [11] and [14]. Our algorithm is based on an induction argument. Suppose that we are given a consistent weighted nilpotent presentation for L/L^l and an epimorphism from L onto L/L^l, we shall show how to extend them to a consistent weighted nilpotent presentation and an epimorphism for L/L^{l+1}.

As computational tools, matrices over $\mathbb{Z}$ play an important rôle. Computing the *row Hermite normal form* is a crucial point of the algorithm. Since its details can be found in [14] and [19], we do not emphasise them further in this paper.

3.1 Computing the Abelian Factor

Suppose that we are given a Lie ring L with finite presentation $\langle \mathcal{X} \mid \mathcal{R} \rangle$, where $\mathcal{X}$ is a finite set of generators, while $\mathcal{R}$ is a finite set of relators. For simplicity, suppose $\mathcal{X} = \{x_1, \ldots, x_n\}$. Set $\mathcal{G} = \{a_1, \ldots, a_n\}$. Construct a map $\phi : \mathcal{X} \to \mathcal{G}$, by simply saying

$$x_i \phi = g_i.$$

Furthermore, we set all PR relations to be trivial and $I = \emptyset$. We call the presentation constructed above a *trivial weighted nilpotent presentation* and it is clear that it represents the free abelian Lie ring $\mathcal{F}_n^{ab}$ on n generators. It is straightforward to see that it contains L/L^2 as a factor ring. Indeed, let $\mathcal{I}$ be the ideal generated by the set $\mathcal{R}\phi$, where we evaluate ϕ as if it was a homomorphism elementwise on $\mathcal{R}$. Then we have

Lemma 3.1 *L/L^2 is isomorphic to $\mathcal{F}_n^{ab}/\mathcal{I}$.*

Proof. Both Lie rings are generated by n elements and satisfy exactly the same relations. It justifies the isomorphic property. □

What we do in practice is the following. Evaluate the relations in $\mathcal{F}_n^{ab}$ as described above and put the images in a matrix M. Then compute the row Hermite normal form M^H for M. It is well known that the rows of M generate the same subgroup, in the free abelian group, as those of M^H. If the first non-zero element, the ith say, of a row of M^H is 1, i.e. we have a row of the form

$$(0, \ldots, 0, 1, m_{i+1}, \ldots, m_n)$$

we simply throw a_i out of $\mathcal{G}$ and modify the map ϕ

$$x_i\phi = -m_{i+1} \cdot a_{i+1} - \cdots - m_n \cdot a_n.$$

Otherwise, if the leading element is greater then one, i.e. we have a row of the form

$$(0, \ldots, 0, m_i > 1, m_{i+1}, \ldots, m_n),$$

we introduce a TR relation

$$m_i \cdot a_i = -m_{i+1} \cdot a_{i+1} - \cdots - m_n \cdot a_n.$$

and put i into I.

Proceeding through each row as described above, we obtain a presentation for L/L^2. We set the surviving generators of weight one and it is straightforward to see that the presentation obtained so far is indeed a presentation for the abelian factor and ϕ extends to an epimorphism. We summarise this result in

Proposition 3.1 *The so obtained presentation is a consistent nilpotent presentation for L/L^2 and ϕ extends to a epimorphism from L onto L/L^2.*

Remark 3.1 *The untouched images under epimorphism will be referred to as definitions of generators of weight one. This terminology will be important later on.*

3.2 Extending the Presentation

In this section we shall describe how to extend the weighted consistent nilpotent presentation of L/L^l to a weighted consistent nilpotent presentation of L/L^{l+1} and how to lift the epimorphism from L onto L/L^l to an epimorphism from L onto L/L^{l+1}. Then the weighted nilpotent presentation can be built for an arbitrary nilpotent factor of L by building it for the abelian factor and iterating the extension finitely many times. Suppose that we have a weighted nilpotent presentation for the factor ring L/L^l, an epimorphism from L onto L/L^l. Extending the presentation will consist of three steps, i.e.

1. extending the epimorphism ϕ,
2. modifying the TR relations,
3. modifying the PR relations.

Extending the epimorphism means introducing a new generator for all $x_i \in \mathcal{X}$, where $x_i\phi$ is not a definition (see Remark 3.1). In other words, if $x_i\phi = w_i$, we modify $x_i\phi$ so that $x_i\phi = w_i + t_i$. The torsion relations will change so that if $i \in I$, $\gamma_i \cdot a_i = w_{ii}$, then we alter $\gamma_i \cdot a_i = w_{ii} + t_{ii}$. Those of the PR relations that are not definitions are modified in a similar way. If $[a_j, a_i] = w_{ji}$ $j > i$ and $[a_j, a_i]$ is not a definition then let the new relation be $[a_j, a_i] = w_{ji} + t_{ji}$.

Note, that all newly introduced generators t_i, t_{ii}, t_{ji} are different from one another. We introduce PR relations so that all of them are central. At this stage we do not alter I. We prove the following.

Theorem 3.1 *The Lie ring defined by the extended presentation contains L/L^{l+1} as a factor ring.*

Proof. Let L/L^l be generated by d generators. Write down the nilpotent presentation for L/L^l and using the definitions of the generators we can write down a presentation for L/L^l consisting only of generators of weight one. Thus we obtained a presentation of the form F/R where F is the free Lie ring on d generators and R is a suitable ideal. Doing the same we get the presentation for L^* as F/R^*. One easily checks that $R^* = [R, F]$.

Let us prove that $F/[R, F]$ contains L/L^{l+1} as a factor ring. Note that L/L^l, L/L^{l+1} and L^* can be generated by d generators. Fix a generating set $f_1, \ldots, f_d$ for F. Let θ and ϕ denote the homomorphisms from F onto F/R and form L/L^{l+1} onto L/L^l, respectively. Note that ϕ has kernel L^l/L^{l+1}. Let $h_1, \ldots, h_d$ be elements of L/L^l such that $h_i\phi = f_i\theta$. L/L^{l+1} is generated by $\{h_1, \ldots, h_d\}$ therefore the map $\psi : f_i \mapsto h_i$ extends to a homomorphism from F onto L/L^{l+1}. Let M be the kernel of ψ. Then it is enough to show that R^* is contained in M. But this is true since, $\psi\phi = \theta$ and $r \in R$ is mapped to the kernel of ϕ by ψ. But the kernel of ϕ is central, hence an element from $[R, F]$ is mapped to zero by ψ. Thus $R^* = [R, F] \leq \ker\psi = M$. □

In practice we use Lemma 2.1. This lemma essentially says, that we do not need to introduce new generators for all PR relations, but only for those of the form $[a_j, a_i] = w_{ji}$ for $j > i$ and a_i is of weight one. The remaining ones can be computed as we have seen in the proof of this lemma. This computation is usually referred to as *computing the tails*.

3.3 Enforcing Consistency

Recall that we called a weighted nilpotent presentation consistent if each element in L uniquely had a normal form. The extended presentation investigated in the previous subsection might not happen to be consistent, as is the case in general.

In order to investigate the uniqueness property we introduce an operation on the nilpotent quotient. First of all let A be an abelian group generated by $\mathcal{G}$ and the TR relations of the extended presentation. Introduce a map ψ such that,

$$\psi : \mathcal{G} \times \mathcal{G} \to A \qquad \psi(a_j, a_i) = \begin{cases} w_{ji} & \text{if } j > i, \\ -w_{ji} & \text{if } j < i, \\ 0 & \text{if } j = i, \end{cases}$$

provided that the weighted nilpotent presentation possesses a PR relation of the form $[a_j, a_i] = w_{ji}$ for $j > i$. Then the following is true.

Proposition 3.2 *ψ can be extended to a bilinear operation on A if and only if*

$$\gamma_j \cdot [a_j, a_i] = \sum_{k=j+1}^{n} \alpha_{jk}[a_k, a_i], \qquad C1$$

where $j \in I$ and the relation corresponding to a_j is of the form

$$\gamma_j \cdot a_j = \alpha_{j,j+1}a_{j+1} + \cdots + \alpha_{in} \cdot a_n.$$

Proof. It is immediate that the condition is necessary. To see that it is sufficient as well, observe that the two expressions, w and w' say, are equal, it means that they can be transformed to each other by using TR relations. The condition of the proposition claims that such a transition respects the equality $\psi(w, v) = \psi(w', v)$ for any v. Similar result is obtained in the second variable of ψ by swapping the two variables. □

If A is equipped by that operation then it happens to be a non-associative ring. We prove the following.

Lemma 3.2 *The weighted nilpotent presentation of L is consistent if and only if A is a Lie ring.*

Proof. Observe that $L = A/J$ where J is the ideal generated by all instances of the Jacobi identity. A normal word ℓ represents the zero element in L if and only if $\ell \in J$. If A is a Lie ring then $J = 0$, so ℓ must be the trivial word. On the other hand, if the nilpotent presentation is consistent, then all instances of the Jacobi identity collect to the trivial word, so $J = 0$ and $A = L$ holds. □

One can easily check the Jacobi identity, since it is a multi-linear identity, thus it is enough to check it on the triples formed by the abelian group generators. However, we can restrict ourselves even more by the following.

Lemma 3.3 *A weighted nilpotent presentation $(\mathcal{G}, S)$ of a Lie ring L is consistent if and only if the relations (C1) and the Jacobi identity holds for each triple $(a_i, a_j, a_k) \in \mathcal{G} \times \mathcal{G} \times \mathcal{G}$ where $1 \leq i < j < k \leq n$ and a_i is of weight one. In other words*

$$[a_i, a_j, a_k] + [a_j, a_k, a_i] + [a_k, a_i, a_j] = 0 \qquad C2$$

for $1 \leq i < j < k \leq n$ and a_i is of weight one.

The above lemma first appeared in [22] for pc-presentation of p-groups. It was modified for Lie algebras in [11], where one finds its proof. The Lie ring case can be handled in the same way as Lie algebras.

3.4 Enforcing the Defining Relations

One can easily observe that the presentation obtained so far is a presentation of the freest Lie ring of class l on the same number of generators as L/L^l that happens to contain L/L^{l+1} as a factor ring in it. What is still needed is to assure that the images of the elements of $\mathcal{R}$, under the epimorphism, are 0. It simply means taking the factor ring over the ideal generated by the epimorphic images.

Let L be freely presented as F/R then one sees that $L/L^l = F/(R+F^l) = F/S$. Then the extended Lie ring L^* was obtained as $L^* = F/S^*$ where

$$S^* = [S, F] = [R + F^l, F] = [R, F] + F^{l+1}.$$

But F/S^* is not necessarily a factor of L. Hence we have to pass to the largest factor of F/S^* which is a factor of F/R too. Obviously it is $F/(R + S^*)$.

$$R + S^* = R + [R, F] + F^{l+1} = R + F^{l+1}.$$

Therefore $L/L^{l+1} = F/(R + S^*) = F/(R + F^{l+1})$.

By the induction hypothesis images of R vanish in L/L^l, so they must lie in L^l/L^{l+1} and we are allowed to use abelian group methods again, such as Hermite normal form and other integer based strategies.

In practice we proceed as follows. Put all elements obtained from the consistency relations (C1), (C2) and the defining relations together in a matrix M. Compute the row Hermite normal form M^H for M. Using those rows of M^H whose leading element is equal to 1, we can eliminate some of the generators from the weighted nilpotent presentation. The other rows express linear dependence among the generators over $\mathbb{Z}$. They can be viewed as TR relations and are added to the presentation. Extend the weight function to the newly introduced generators, by saying that they are of weight i and add the indices of generators that were provided with TR relations to I. The following lemma makes sure that generators of weight i have definitions.

Lemma 3.4 *If L is a Lie ring and suppose that L/L^2 is additively generated as*

$$L/L^2 = \langle \ell_1 + L^2, \ldots, \ell_k + L^2 \rangle$$

and L^{i-1}/L^i is additively generated as

$$L^{i-1}/L^i = \langle \ell_1{}' + L^i, \ldots, \ell_l{}' + L^i \rangle$$

then L^i/L^{i+1} is additively generated as

$$L^i/L^{i+1} = \langle [\ell'_m, \ell_n] + L^{i+1} \mid 1 \leq m \leq l,\ 1 \leq n \leq k \rangle.$$

The proof of the above lemma can be done essentially in the same way as in the group case. That proof can be found in [19] Proposition 2.6. But see also [18] for a detailed study of the Lie ring case.

4 The Graded Case

4.1 *Simplifying* LieNQ *in Graded Lie Rings*

The following section is devoted to graded Lie rings. Gradedness of a Lie ring is defined in accordance with the following definition.

Definition 4.1 *Let L be a Lie ring, L is said to be graded if L, as an abelian group, splits into a direct sum of its lower central factors.*

Since a Lie ring of that kind possesses a relatively easy structure, we are allowed to simplify our algorithm in this case.

First of all, we need not extend the TR relations any more since if $a_i \in L^l/L^{l+1}$ for some $i \in I$ implies $\gamma_i \cdot a_i \in L^l/L^{l+1}$, so the right-hand side of the relation corresponding to a_i cannot contain generators of weight greater than l.

Similar thing happens when we extend the PR relations. At the lth step we only alter the relations of the form

$$[a_j, a_i] = w_{ji} \qquad \text{for } i > j \quad i + j = l.$$

We regard these facts when we introduce new generators and compute the tails.

Now it should be clear that enforcing the consistency simply means enforcing the consistency relations (C2) for the triples $(a_i, a_j, a_k) \in \mathcal{G} \times \mathcal{G} \times \mathcal{G}$, where $1 \leq i < j < k \leq n$ $\omega(a_i)+\omega(a_j)+\omega(a_k) = l$ and $\omega(a_i) = 1$. And the relations of type (C1) are checked for only pairs $(a_i, a_j) \in \mathcal{G} \times \mathcal{G}$, where $\omega(a_i) + \omega(a_j) = l$. Note that l still denotes the nilpotency class of the current factor.

We are allowed to simplify it further, namely we need to enforce only those relations of $\mathcal{R}$ that have weight l.

The natural question arises. How can one recognise a graded Lie ring by looking at its finite presentation? We only mention here the well known fact that a Lie ring defined by homogeneous relations is always graded.

4.2 Another approach – Canonical Weighted Nilpotent Presentation

In a graded Lie ring we shall construct the so-called *canonical nilpotent presentation* instead of the weighted nilpotent presentation defined in Section 2. The concept of the canonical weighted nilpotent presentation fits the concept of canonical presentation for finitely generated abelian groups. In the sequel we shall use additive notation for abelian groups and $+$ to denote the binary operation in such structures. Recall that C_n denotes the cyclic group of order n.

Theorem 4.1 *Let $(A, +)$ be a finitely generated abelian group. Then A can be written as a direct sum of cyclic groups, i.e.*

$$A \cong C_{k_1} \oplus \cdots \oplus C_{k_r} \oplus C_\infty \oplus \cdots \oplus C_\infty, \tag{9}$$

where $k_1|k_2, \ldots, k_{r-1}|k_r$. The (9) form is called the canonical decomposition for A. The canonical decomposition is unique for an arbitrary such A.

The proof of the theorem can be found in [21] Theorem 5.2.

In the following we define the canonical weighted nilpotent presentation for a graded Lie ring.

Definition 4.2 *A Lie ring presentation is said to be canonical weighted nilpotent presentation, if it is of the form*

$$\langle a_1, \ldots, a_n \mid c_i \cdot a_i \quad \textit{for } i \in I,\ [a_j, a_i] = w_{ij} \quad \textit{for } j > i \rangle, \tag{10}$$

where the w_{ij} are normal words as in (7).

Remark 4.1 *Using the results of Section 3 we shall see that each finitely presented graded nilpotent Lie ring possesses such a presentation. Essentially it is enough to see, that the generators corresponding to a certain lower central factor can be chosen so that the torsion relations are trivial in (10). Conversely, it is rather easy to see that a Lie ring defined by a canonical weighted nilpotent presentation is always a finitely presented, nilpotent and graded.*

Remark 4.2 *Actually we shall construct more than a simple presentation defined in (10). We shall construct a presentation in which the lower central factors are canonically presented as abelian groups, according to Theorem 4.1. That is, we shall enforce the appropriate divisibility conditions for the c_i.*

The following theorem is an easy consequence of Theorem 4.1 and Definition 4.2.

Theorem 4.2 *Requiring the divisibility condition for each lower central factor in (10), the coefficients γ_i are uniquely determined by the isomorphism type of L.*

4.3 Computing the Canonical Weighted Nilpotent Presentation

After this preparation we are ready to describe, how one can compute the canonical weighted nilpotent presentation for a finitely presented nilpotent graded Lie ring.

Recall that we computed the matrix consisting of consistency relations and defining relations. We computed its Hermite normal form. Now we shall compute its Smith normal form. Recall that if M is a $n \times m$ matrix then there exist $n \times n$ and $m \times m$ matrices P and Q, respectively, such that $PMQ = S$ where S is the Smith normal form of M. To construct canonical nilpotent presentation we shall compute S, Q and Q^{-1}. Using Proposition 8.3.1 of [19] one can compute new generators for the canonical presentation. Put $Q = (q_{i,j})$ and $Q^{-1} = (\hat{q}_{i,j})$.

The condition for the existence of the definition cannot be kept any more in that case. We use the following approach instead.

Suppose that at a certain layer we introduced new generators $x_1, \ldots, x_s$, such that they correspond to the following products:

$$x_i = [b_{1i}, b_{2i}] \quad \text{where} \quad \omega(b_{1i}) = 1 \quad \text{and} \quad \omega(b_{2i}) = \omega(x_1) - 1, \ 1 \leq i \leq s.$$

Using Proposition 8.3.1 of [19] one gets the generators $a_1, \ldots, a_s$ of the canonical presentation for that layer. Moreover one has that

$$\begin{pmatrix} a_1 \\ \vdots \\ a_n \end{pmatrix} = Q^{-1} \begin{pmatrix} [b_{11}, b_{21}] \\ \vdots \\ [b_{1s}, b_{2s}] \end{pmatrix}.$$

So, we can introduce definitions for the canonical generators of the following form

$$a_i := \hat{q}_{i,1}[b_{11}, b_{21}] + \cdots + \hat{q}_{i,s}[b_{1s}, b_{2s}] \qquad \text{for } 1 \leq i \leq s.$$

That is, the definitions are linear combinations consisting of Lie products with generators of smaller weight.

In Section 2 we already computed products of current weight. They need one more look as follows. After tail computation, we expressed those products in terms of newly introduced generators $x_1, \ldots, x_s$.

Now, those generators are to be replaced by the canonical generators $a_1, \ldots, a_s$. To do that we use Proposition 8.3.1 of [19] again. That is,

$$x_i = q_{i,1}a_i + \cdots + q_{i,s}a_s \qquad \text{for } 1 \leq i \leq s.$$

We proceed the old generators occuring in the epimorphic images in the same way.

Lemma 2.1 is still true in the canonical case and its proof can be done in essentially the same way as presented before. Lemma 3.3 seemingly has no connection to the form of the definitions. However, having a deeper look at its proof, one finds that it depends heavily on the concept of definitions, but can be modified to the new approach too.

The canonical weighted nilpotent presentation computed so far depends on the transformer unimodular matrix Q. Among many possible matrices, we want to choose the one with the smallest possible entries, because they will appear in the presentation. This is a hard problem as is stated next.

Theorem 4.3 *Let L be a finitely presented nilpotent, graded Lie ring. The task of minimising the vectors in the right-hand side of the epimorphic images and the PR relations, according to the maximum norm, (MINLIE) is NP-hard.*

Proof. Let MINGCD be the problem that is proved to be NP-hard by Theorem 2 in [8]. We reduce MINGCD to MINLIE in polynomial time. Let us be given an instance $(c_1, \ldots, c_n)$ of MINGCD. Write down the finite presentation

$$\langle x_1, \ldots, x_n \mid c_1x_1 + \cdots + c_nx_n = 0,\ [x_i, x_j] = 0 \quad \text{for } 1 \leq i < j \leq n \rangle.$$

Clearly, this can be done in time proportional to n^2. Use LIENQ on this presentation. LIENQ terminates at the abelian factor and produces the following output:

$$\begin{aligned} x_i\varphi = q_{i,1}a_1 + \cdots q_{i,n}a_n \qquad & \text{for } 1 \leq i \leq n, \\ \gcd(c_1, \ldots, c_n)a_1 = 0, & \\ [a_i, a_j] = 0 \qquad & \text{for } 1 \leq j < i \leq n. \end{aligned} \tag{11}$$

Where the transformer matrix $Q = (q_{i,j})$. The ability of minimising the vectors in (11) implies the ability of minimising the vectors in Q and hence solving MINGCD. □

Another disadvantage of this approach is presented by the following theorem.

Theorem 4.4 *The absolute values of the coefficients appearing in the canonical nilpotent presentation cannot be bounded by a function depending only on the number of generators.*

Proof. Easy consequence of [8] Lemma 4. □

5 Performance

5.1 *Implementing* LieNQ *in C*

The implementation for the LIENQ algorithm has been written in the C programming language. The currently available version is Version 2.0, which provides time measure, and computations for graded and non-graded cases separately.

The program takes a finite presentation $(\mathcal{X}, \mathcal{R})$ for a Lie ring L and the nilpotency class of required factor as its input. It outputs the TR, PR relations and the images of the elements of $\mathcal{X}$ under the epimorphism. The nilpotency class can be omitted, in this case the program runs until the lower central series stabilises or the algorithm goes beyond the capacity of the computer.

Throughout the computation the data of the weighted nilpotent presentation is stored in normal word form. At previous stages they were stored as coefficient vectors, but it turned out to be more efficient in an array consisting of structures, where the first element of a structure contains the number of a generator and the second one its coefficient (the idea is due to [15]). For instance, the Lie ring element

$$0 \cdot a_1 + 0 \cdot a_2 + 2 \cdot a_3 - 4 \cdot a_4 + 0 \cdot a_5$$

is stored as

$$\big((3, 2), (4, -4), (0, 0)\big).$$

The first component of the last pair indicates an `EOW` (end-of-word) sign, while the second one, in fact, is arbitrary.

The information of the weighted nilpotent presentation is stored in several arrays as follows:

`Coefficients[i]`	γ_i if $i \in I$, 0 otherwise,
`Power[i]`	w_i, right-hand side of the TR relation $\gamma_i \cdot a_i$,
`Product[j][i]`	w_{ji}, right-hand side of the PR relation $[a_j, a_i]$,
`Epimorphism[i]`	$x_i\phi$, the ith epimorphic image,
`Weight[i]`	$\omega(a_i)$, the weight of a_i,
`Dimension[i]`	number of generators of weight i,
`Definition[i]`	the definition of a_i.

Note that `Definition[i]` is an array of structures. A component of the array either contains the number of two generators and a positive integer, if the definition of a_i is like $a_i = \sum c_i \cdot [a_{k_i}, a_{l_i}]$, or contains the number of the generator from $\mathcal{X}$ in the first component and 0 in the second and the third if a_i is defined as an epimorphic image. In the latter case the length of the array is, of course, one.

The defining relations are stored as expression trees. See [14] for its description which emphasises its advantages in details.

During the computation graded and non-graded cases are distinguished according to whether the user switched on the `-g` option or not. There are different tails routines, that is `Tails` and `GradedTails`, and consistency routines, like `Consistency` and `GradedConsistency`, built into the code. Of course, introducing new generators is also a different task.

It is, however, important to introduce the new generators in the right order. Lemma 3.4 makes it possible to express all tails in terms of generators a_k of weight l where $a_k = [a_j, a_i]$ for some a_j of weight $l - 1$ and a_i of weight 1. In order to use this fact in practice, we introduce those generators last and when we compute the row Hermite normal form for the matrix they will be, in fact, the only surviving generators and all the others disappear. In this way, we ensure that the new generators indeed have definitions, so the restrictions we kept are reasonable. In the graded case such restrictions are not kept.

The main routines for matrix computation and the basics of the parser program are due to W. Nickel. The first mentioned one uses the GNU MP package to deal with long integers.

LIENQ was successfully compiled on several UN*X platforms, such that Linux, Free-BSD, SunOS, Solaris.

5.2 GAP Interface

LIENQ might be invoked from GAP [17] using a GAP interface. Since GAP does not know about Lie algebras until version 3.5, it is necessary to have at least GAP-3.5 to use the interface. We remark, that even GAP-3.5 knows only *Lie algebra*s but not Lie rings over a ring, so LIENQ kills torsions in that case and GAP considers the result as Lie algebra over $\mathbb{Q}$.

6 Some Sample Computations

In this section we present some examples, that were tried by LIENQ. We consider the following Lie ring presentations.

$$
\begin{aligned}
L_1 &= \langle x, y \rangle \\
L_2 &= \langle x, y \mid [y,x,y], [y,x,x,x,x,x] \rangle \\
L_3 &= \langle x, y \mid |[[y,x],x,y], [[y,x],y,x], [[y,x],x,x] + [[y,x],y,y] \rangle \\
L_4 &= \langle a, b, c, d, e \mid [b,a], [c,a], [e,c], [e,d], [d,a] = [c,b], [d,b] = [e,a], [d,c] = [e,b] \rangle
\end{aligned}
$$

$$
\begin{aligned}
L_5 = \langle & e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10} \mid \\
& [e_2,e_1,e_1], [e_1,e_2,e_2], [e_3,e_1], [e_4,e_1], [e_5,e_1], [e_6,e_1], [e_7,e_1], [e_8,e_1], \\
& [e_9,e_1], [e_{10},e_1], [e_3,e_2,e_2], [e_2,e_3,e_3], [e_4,e_2], [e_5,e_2], [e_6,e_2], [e_7,e_2], \\
& [e_8,e_2], [e_9,e_2], [e_{10},e_2], [e_4,e_3,e_3], [e_3,e_4,e_4], [e_5,e_3], [e_6,e_3], [e_7,e_3], \\
& [e_8,e_3], [e_9,e_3], [e_{10},e_3], [e_5,e_4,e_4], [e_4,e_5,e_5], [e_6,e_4], [e_7,e_4], [e_8,e_4], \\
& [e_9,e_4], [e_{10},e_4], [e_6,e_5,e_5], [e_5,e_6,e_6], [e_7,e_5], [e_8,e_5], [e_9,e_5], [e_{10},e_5], \\
& [e_7,e_6,e_6], [e_6,e_7,e_7], [e_8,e_6], [e_9,e_6], [e_{10},e_6], [e_8,e_7,e_7], [e_7,e_8,e_8], \\
& [e_9,e_7], [e_{10},e_7], [e_9,e_8,e_8], [e_8,e_9,e_9], [e_{10},e_8], [e_{10},e_9,e_9], [e_9,e_{10},e_{10}], \\
& [e_3,e_2,e_1,e_2], [e_4,e_3,e_2,e_3], [e_5,e_4,e_3,e_4], [e_6,e_5,e_4,e_5], [e_7,e_6,e_5,e_6], \\
& [e_8,e_7,e_6,e_7], [e_9,e_8,e_7,e_8], [e_{10},e_9,e_8,e_9] \rangle.
\end{aligned}
$$

The first example presents the free Lie ring of rank two. The second and third examples are found in [1] and are of great importance from the point of view of thin Lie algebras. The fourth example is due to Jürgen Wisliceny and has a very nice structure, as we shall see later. The fifth one is not else than the presentation of the positive part of the classical Lie algebra A_{10} factored out by the ideal generated by the torsions appearing over $\mathbb{Z}$.

The structure of L_4 is very nice as we mentioned before. Having a look at its canonical weighted nilpotent presentation one has the following.

Theorem 6.1 *The underlying abelian group of the lower central factors of L_4 is of the form*

$$
L^{i+1}/L^{i+2} \cong \begin{cases} C_{i/2} \oplus C_\infty \oplus C_\infty \oplus C_\infty & \text{if } i \text{ is even}, \\ C_\infty \oplus C_\infty \oplus C_\infty \oplus C_\infty \oplus C_\infty & \text{if } i \text{ is odd}. \end{cases}
$$

for $0 \le i \le 67$.

By computation, the following structure theorem is true for L_5.

Theorem 6.2 *L_5 is nilpotent of class 10. Its lower central factors are of the form*

$$L_5^i/L_5^{i+1} = C_\infty \oplus \cdots \oplus C_\infty, \tag{12}$$

where the number if direct factors is $11-i$ for $1 \leq i \leq 10$.

The structure of L_1 is well known. Witt's formula ([5] Theorem 11.2.2) can be used to compute the dimension for a lower central factor of L_1. The result of the computation coincides with the known dimensions.

The following table contains the computational informations concerning those examples.

example	class computed	nr. of gens.	torsion rank	CPU time (ms)
L_1	10	226	0	811100
L_2	14	89	70	66200
L_3	12	81	63	33150
L_4	7	51	22	41333
L_5	6	45	0	254016

Since all Lie rings presented by those relations are graded we used the graded approach to compute the presentation for them. That is summarised in the following table.

example	class computed	nr. of gens.	torsion rank	CPU time (ms)
L_1	10	226	0	8633
L_1	12	757	0	257383
L_2	14	57	38	3150
L_2	18	195	171	197483
L_3	12	81	63	4133
L_4	67	300	32	903553
L_5	10	55	0	36083

All computations, except for the 67th class of L_4, were done on a 486-DX4 100 MHz PC with 16 MB memory plus 16 MB swap space. The above mentioned exception was computed on a Sun SparcCenter 2000 with 256 MB of RAM.

One easily sees that using those ideas speeds up the computation by a significant factor. The need of using a better algorithm for Smith normal form computation is reasonable. While computing L_3 an integer overflow occured with 32 bit arithmetic at the 68th factor.

7 Acknowledgements

The author whishes to express his gratitude to those who helped him in the preparation of this paper, but especially to Dr Werner Nickel.

References

[1] A. E. Caranti, Sandro Mattarei, M. F. Newman, and C. M. Scoppola. Thin groups of prime-power order and thin Lie algebras. Manuscript, 1994.

[2] Frank Celler, M.F. Newman, Werner Nickel, and Alice C. Niemeyer. An algorithm for computing quotients of prime-power order for finitely-presented groups and its implementation in GAP. Technical Report SMS-127-93, School of Mathematical Sciences, Australian National University, 1993.

[3] Martin D. Davis and Elaine J. Weyuker. *Computability, Complexity and Languages*. Academic Press, 1983.

[4] Philip Hall. Some word problems. *Journal of London Math. Soc.*, 33:482–496, 1958.

[5] Marshall Hall, Jr. *The Theory of Groups*. Macmillan Co., New York, 1959.

[6] B. Hartley and T.O. Hawkes. *Rings, Modules and Linear Algebra*. Chapman and Hall, London, 1970.

[7] George Havas, Derek F. Holt, and Sarah Rees. Recognizing badly presented Z-modules. *Linear Algebra Appl.*, 192:137–163, 1993.

[8] George Havas and Bohdan S. Majewski. Hermite normal form computation for integer matrices. *Congressus Numerantium*, 105:87–96, 1994.

[9] George Havas and Bohdan S. Majewski. Integer matrix diagonalization. *J. Symbolic Comput.*, 11, 1995.

[10] George Havas and M.F. Newman. Application of computers to questions like those of Burnside. In *Burnside Groups*, volume 806 of *Lecture Notes in Math.*, pages 211–230, Berlin, Heidelberg, New York, 1980. (Bielefeld, 1977), Springer-Verlag.

[11] George Havas, M.F. Newman, and M.R. Vaughan-Lee. A nilpotent quotient algorithm for graded lie rings. *J. Symbolic Comput.*, 9:653–664, 1990.

[12] J.E Humphreys. *Introduction to Lie Algebras and Representation Theory*. Graduate Text in Mathematics. Springer-Verlag New York, Heidelberg, Berlin, 1972.

[13] Werner Nickel. *Central extensions of polycyclic groups*. PhD thesis, Australian National University, 1993.

[14] Werner Nickel. Computing nilpotent quotients in finitely presented groups. In *Geometric and Computational Prospectives of Finite Groups*, DIMACS series, 1995. To appear.

[15] Werner Nickel. Personal communication, 1995.

[16] Derek J. Robinson. *A Course in the Theory of Groups*, volume 80 of *Graduate Texts in Math.* Springer-Verlag, New York, Heidelberg, Berlin, 1982.

[17] Martin Schönert *et al.* GAP – *Groups, Algorithms and Programming*. Lehrstuhl D für Mathematik, RWTH, Aachen, 1994.

[18] Csaba Schneider. Computations in finitely presented lie rings. Master's thesis, Kossuth Lajos University of Arts and Sciences, 1996.

[19] Charles C. Sims. *Computation with finitely presented groups*. Cambridge University Press, 1994.

[20] Henry J.S. Smith. On systems of linear indeterminate equations and congruences. *Philos. Trans. Royal Soc. London*, cli:293–326, 1861.

[21] Michio Suzuki. *Group Theory I*, volume 247 of *Grundlehren Math. Wiss.* Springer-Verlag, Berlin, Heidelberg, New York, 1982.

[22] M.R. Vaughan-Lee. An aspect of the nilpotent quotient algorithm. In *Computational Group Theory*, pages 76–83, London, New York, 1984. (Durham, 1982), Academic Press.