Testing membership in unitriangular matrix groups
Preliminary draft

Gábor Ivanyos *

January 2, 1996

Abstract

We present an algorithm that, given a subgroup G of the group of the upper triangular matrices with rational entries by a set of generators, computes generators of the lower central series of G. As an application, we show that the constructive membership problem in G can be performed in polynomial time. This draft is intended to be part of a larger project investigating the complexity of computational problems in finitely generated linear groups.

1 Introduction

Polynomial time algorithms for computational problems in some classes of matrix groups over algebraic number fields have been recently obtained. The *constructive membership problem* in a linear group G given by a finite set of generators has been shown to be soluble in polynomial time in the one-dimensional case by Ge [Ge1, Ge2], in the abelian case by Cai, Lipton, Zalcstein [CLZ], Babai, Beals, Cai, Ivanyos, and Luks [BBCIL], in the abelian-by-finite case by Beals [Be]. Beals [Be] also presents a polynomial time algorithm that decides whether G is soluble-by-finite (nilpotent-by-finite, or abelian-by-finite) and if this is the case then computes *normal generators* of a soluble (nilpotent, abelian, respectively) normal subgroup of finite index.

Ostheimer [Os] has developed similar algorithms independently. In the polycyclic-by-finite case, based on computing a *polycyclic presentation* of the maximal unipotent normal subgroup, she also presents (without complexity analysis) an algorithm for the membership problem.

* Computer and Automation Institute, Hungarian Academy of Sciences, 1111 Budapest, Lágymányosi u. 11, Hungary e-mail: `ig@ilab.sztaki.hu` This work was done while the author was visiting SUNY at Buffalo, Dept. of Computer Science. Research supported in part by NSF grants CCR-9057486 and CCR-9319093, and the Alfred P. Sloan Foundation.

Most difficulties in analyzing complexity arise in *iterative* computations in unipotent groups. A polynomial time algorithm to compute the lower central series of a unitriangular group. The author is convinced that the result leads to a polynomial time algorithm for the constructive membership problem in the finite-by-polycyclic case. There is also some hope that the method can be useful in studying the complexity of the membership problem in the soluble-by-finite case.

Since this draft is intended to be a fragment of larger work, the detailed introduction is omitted. The reader is referred to [CLZ] [BBCIL], or [Be] for background and basic concepts. We use very elementary tools form the theory of nilpotent Lie groups and algebras. We use very elementary tools form the theory of nilpotent Lie groups and algebras (c.f. [FV] and [Po]).

2 The algorithm

Assume that we are given $G \leq UT_n(\mathbf{Q})$ by a generating system $S = \{s_1, \ldots, s_m\} \subseteq UT_n(\mathbf{Q})$, where UT_n stands for group of the upper triangular unipotent matrices. Let $N_n(\mathbf{Q})$ denote the set of upper triangular nilpotent matrices. $N_n(\mathbf{Q})$ is in fact a nilpotent (associative) subalgebra in $M_n(\mathbf{Q})$. It follows that $N_n(\mathbf{Q})$ is a nilpotent Lie algebra with the bracket operator $[x, y] = xy - yx$.

Using elementary methods from Lie theory, we compute the lower central series of G. For convenience of the reader we attempt to be more or less self-contained. In our case case we do not have to use topological arguments, and we can perform computations over the rationals. The only tools used are the mapping exp, its inverse log defined by formal power series, and the Campbell-Hausdorff series reflecting the most important properties. For $x \in N_n(\mathbf{Q})$,

$$\exp x = \sum_{i=0}^{\infty} \frac{1}{i!} x^i = \sum_{i=0}^{n-1} \frac{1}{i!} x^i,$$

and for $x \in UT_n(\mathbf{Q})$,

$$\exp x = \sum_{i=1}^{\infty} \frac{(-1)^{i-1}}{i} (x-1)^i = \sum_{i=1}^{n-1} \frac{(-1)^{i-1}}{i} (x-1)^i.$$

For every $x, y \in UT_n(\mathbf{Q})$ the log of xy can be expressed by the Campbell-Hausdorff series

$$\log(xy) = \log x + \log y + \frac{1}{2}[\log x, \log y] + \sum_{i=3}^{n-1} \mathbf{D}_i(\log x, \log y),$$

where for every i, $\mathbf{D}_i(u, v)$ is a homogeneous Lie polynomial over $\mathbf{Q}$ in u and v of degree i (the Dynkin polynomials). Recall that a Lie monomial in u, v of degree $i > 1$ is a term $[[[u_1, u_2], \ldots], u_i]$, where $u_1, u_2, \ldots, u_i \in \{u, v\}$. (The unique Lie monomial of degree 1 is $x + y$.) A homogeneous Lie polynomial over $\mathbf{Q}$ of degree i is a $\mathbf{Q}$-linear combination of Lie monomials of degree i.

We start with computing (a basis of) $L_1 = L = L(G)$, the rational Lie algebra generated by $\log S = \{\log s_1, \ldots, \log s_n\}$. By the Campbell-Hausdorff formula, L is in fact the Lie algebra over $\mathbf{Q}$ generated by $\log G = \{\log g | g \in G\}$. We also compute $L_2 = [L, L], \ldots, L_r = [L_{r-1}, L_1], L_{r+1} = [L_r, L_1] = 0$, the lower central series of L_1. (In the traditional notation $L_i = L^{i-1}$. We use this indexing to express the correspondence with degrees of Lie polynomials.) For $i = 1, \ldots, r$, let $\mathcal{G}_i = \{\exp x | x \in L_i\}$. Note that $\mathcal{G} = \mathcal{G}_1$ consists of the rational points of the (real or complex) Lie group corresponding the (real or complex) Lie algebra obtained from L by extending scalars. What follows is the well known correspondence between the central series of Lie groups and their Lie algebras.

Again by the C-H formula, $\mathcal{G}_i \leq UT_n(\mathbf{Q})$ and we have that $\log : \mathcal{G}_1 \to L$ is a bijection (the inverse of exp). For the logarithm of products and commutators we have:

$$\log(xy) = \log x + \log y + \frac{1}{2}[\log x, \log y] + \text{commutators of degree} \geq 3,$$

and

$$\log(x^{-1}y^{-1}xy) = [\log x, \log y] + \text{commutators of degree} \geq 3.$$

It follows that if $x \in \mathcal{G}_i$ and $y \in \mathcal{G}_j$ then

$$\log(xy) \equiv \log(x) + \log(y) \pmod{L_{i+j}}$$

and

$$\log(x^{-1}y^{-1}xy) \equiv [\log x, \log y] \pmod{L_{i+j+\min\{i,j\}}}.$$

In particular, the restriction of log to $\mathcal{G}_i$ induces an isomorphism $\mathcal{G}_i/\mathcal{G}_{i+1} \cong L_i/L_{i+1}$, the latter is understood as the additive group. Also, it is easy to see that $\mathcal{G}_{i+1} = [\mathcal{G}_i, \mathcal{G}_1]$. In other words, $\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_r$ is the lower central series of $\mathcal{G}$. (In the traditional notation $\mathcal{G}_i = \mathcal{G}^{i-1}$.)

Let $G_i = G \cap \mathcal{G}_i$. It is obvious that $[G_i, G_1] \leq G_{i+1}$.

Now S generates G, and $\log S$ generates L, therefore $\log S$ generates a lattice of full rank in the vector space L/L_2. By lifting back a basis of this lattice to G, we find $S_1 \subseteq G$ such that S_1 (modulo $\mathcal{G}_2$) is a basis of the group $G\mathcal{G}_2/\mathcal{G}_2 \cong G/G_2$. Inductively, assume that the set $S_i \subseteq G_i$ is such that S_i is a basis of the factor group $G_i\mathcal{G}_{i+1}/\mathcal{G}_{i+1} \cong G_i/G_{i+1}$, and, $\log S_i$ is a $\mathbf{Q}$-basis of the vector space L_i/L_{i+1}. It follows that $L_{i+1} = [L_{i+1} + span(\log S_i), L_2 + span(\log S_1)] \subseteq L_{i+2} + span([\log S_i, \log S_1]) = L_{i+2} + span(\log[S_i, S_1])$, therefore $\log[S_i, S_1]$ generates a lattice of full rank in L_{i+1}/L_{i+2}. Lifting a basis of this lattice to G, we obtain a basis $S_{i+1} \subseteq [G_i, G_1]$ of the factor group $G_{i+1}\mathcal{G}_{i+2}/\mathcal{G}_{i+2} \cong G_{i+1}/G_{i+2}$, such that $\log S_{i+1}$ is a vector space basis of $\mathcal{G}_{i+1}/\mathcal{G}_{i+2}$. This argument also shows that $G = G_1 > G_2 > \ldots$ is in fact the lower central series of G.

Finally, we obtain a generating system $S_1 \cup S_2 \cup \ldots S_r$ of G such that for every $1 \leq i \leq r$, $S_i \cup \ldots S_r$ is a generating system of G_i and S_i is a basis of the free abelian group G_i/G_{i+1}.

To keep the size of the basis elements polynomial, in each round, we take S_i such that $\log S_i$ is a reduced basis (in the sense of [LLL]) of the lattice $\log G_i + L_{i+1}/L_{i+1}$. To prove that it works we have to show that the lattice $\log G_i + L_{i+1}/L_{i+1}$ is of polynomial size (i.e., admits a basis of polynomial size). It is enough to prove that $\log G_i + L_{i+1}/L_{i+1}$ is in a "sandwich" of lattices of polynomial size: there exists lattices M_i and N_i of polynomial size in L_i/L_{i+1} such that $M_i \subseteq \log G_i + L_{i+1}/L_{i+1} \subseteq N_i$

The lower bound can be obtained from a simplification of the algorithm to compute S_i. Let $\tilde{S}_1 = S_1$, and in each step, we choose a maximal linearly independent set from $\log[\tilde{S}_i, \tilde{S}_1]$ modulo L_{i+2}. It can be easily shown by induction, that for every i, $M_i = \log \tilde{S}_i + L_{i+1}$ is a sublattice of full rank of $\log G_i + L_{i+1}/L_{i+1}$. On the other hand, every element of S_i is an iterated commutator of i elements from S_1, therefore must be of polynomial size.

The upper bound is not difficult either. Conjugating by an appropriate diagonal matrix, we may achieve the situation where $G \leq UT_n(n!\mathbf{Z})$, i.e, the elements of S are upper triangular unipotent matrices with integer entries divisible by $n!$. Then for every $g \in G$ we have that $\log g$ is in the lattice H of the upper triangular nilpotent matrices with integer entries, therefore $G_i + L_{i+1}/L_{i+1}$ is a sublattice of the lattice $N_i = H + L_{i+1} \cap L_i/L_{i+1}$.

Note that there is an alternative way to find *generators* in $G_i\mathcal{G}_{i+1}/\mathcal{G}_{i+1}$. Namely, if S_i is set of generators of $G_i\mathcal{G}_{i+1}/\mathcal{G}_{i+1}$, then from $[S_i, S_{i+1}]$ we can choose a set of generators of $G_{i+1}\mathcal{G}_{i+2}/\mathcal{G}_{i+2}$. The bounds on the lattice guarantee that we can always choose a set of polynomially many elements. (If we already have a set of full rank, then adding a new element increases the discriminant by a multiplicative factor.)

We have proved

Theorem 1 *Given a finitely generated $G \leq UT_n(\mathbf{Q})$ by generators, we can find in polynomial time $S_1, S_2, \ldots, S_r \subseteq G$ such that for every i, S_i (modulo G^i) is a basis of the factor group G^{i-1}/G^i, where G^{i-1} is the ith element of the lower central series of G.*

Corollary 2 *Given a finitely generated $G \leq UT_n(\mathbf{Q})$ by a set of generator S, and an element $x \in UT_n(\mathbf{Q})$, we can decide in polynomial time whether $x \in G$, and if $x \in G$ we can find find in polynomial time an expression (straight line program) of x in terms of S.*

Proof. We start with testing whether $x \in \mathcal{G}$. If it is, $\log x + L_2$ can be written as a unique linear combination of the elements $\log s_j^1 + L_2$, where s_j^1 are the elements of S_1. The coefficients are integers if and only if $x\mathcal{G}_2 \in G\mathcal{G}_2$. If the coefficients are integers, we divide x by the product of appropriate powers of

s_j^1 and obtain an element (denoted again by x) of $\mathcal{G}_2$, and we can proceed with $G_2, L_2, \mathcal{G}_2$.... By Cramer's rule it is easy to see that the sizes are polynomial.... $\square$

References

[BBCIL] L. Babai, R. Beals, J-y. Cai, G. Ivanyos, E. M. Luks: Multiplicative equations over commuting matrices, *Proc. 7th ACM-SIAM SODA (Symp. on Discrete Algorithms),* 1996, to appear.

[BBR] L. Babai, R. Beals, D. Rockmore: Deciding finiteness of matrix groups in deterministic polynomial time, *Israel J. Math.*, to appear.

[BCRS] G. Baumslag, F. B. Cannonito, D. J. Robinson, D. Segal: The arithmetic theory of polycyclic-by-finite groups, *J. Algebra* **142** (1991), 118-149.

[Be] R. Beals: Algorithms for matrix groups and the Tits alternative, *Proc. 36th IEEE FOCS* (1995), 593–602.

[CLZ] J-y. Cai, R. J. Lipton, Y. Zalcstein: The complexity of the membership problem for 2-generated commutative semigroups of rational matrices, *Proc 35th IEEE FOCS* (1994), 135–142.

[FV] H. Freudenthal, H. de Vries: *Linear Lie groups,* Academic Press 1969.

[Ge1] G. Ge: Algorithms related to multiplicative representations of algebraic numbers, PhD thesis, Math Dept, U. C. Berkeley, 1993.

[Ge2] G. Ge: Testing equalities of multiplicative representations in polynomial time, *Proc 34th IEEE FOCS* (1993), 422–426.

[LLL] A. K. Lenstra, H. W. Lenstra, L. Lovász: Factoring polynomials with rational coefficients, *Math. Ann.* **261** (1982), 515–534.

[Os] G. Ostheimer: Algorithms for polycyclic-by-finite matrix groups DIMACS Technical Report 95-33.

[Po] Postnikov: Lie groups and Lie algebras?

[Si] C. C. Sims: Computation with finitely presented groups, Cambridge University Press 1994.