Complexity of Decision Problems in Polycyclic Groups

Delaram Kahrobaei

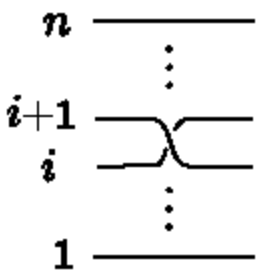
New York City College of Technology
City University of New York
DKahrobaei@Citytech.CUNY.edu
http://websupport1.citytech.cuny.edu/Faculty/dkahrobaei/

Cryptography and Combinatorial Group Theory Seminar
Graduate Center
Cryptography Sub Seminar

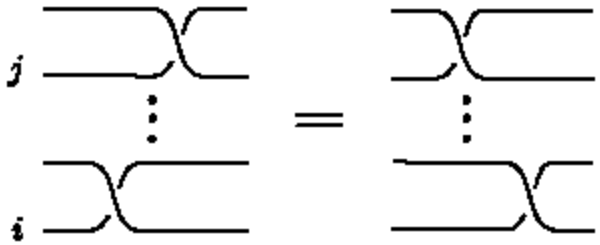
February 15th 2008

Braid Group Presentation

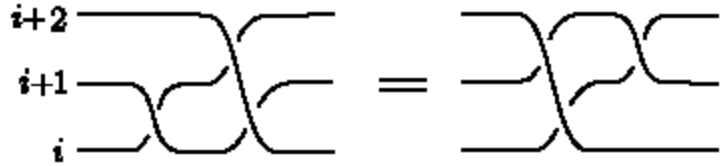
$$B_n = \left\langle \sigma_1, \ldots, \sigma_{n-1} \;\middle|\; \begin{array}{l} \sigma_i\sigma_j = \sigma_j\sigma_i, \quad |i-j| \geq 2 \\ \sigma_i\sigma_{i+1}\sigma_i = \sigma_{i+1}\sigma_i\sigma_{i+1}, \quad i = 1, \ldots, n-2 \end{array} \right\rangle$$



σ_i



$\sigma_i\sigma_j = \sigma_j\sigma_i$



$\sigma_i\sigma_{i+1}\sigma_i = \sigma_{i+1}\sigma_i\sigma_{i+1}$

Polycyclic Groups

A group is called **polycyclic** if there exists a polycyclic series through the group; i.e. a subnormal series of finite length with cyclic factors $1 = G_0 \trianglelefteq G_1 \trianglelefteq \ldots \trianglelefteq G_n = G$

G_{i+1}/G_i is cyclic

Polycyclic Presentation

$$P = \langle a_1, \ldots, a_n \mid a_j^{a_i} = w_{ij},\ a_j^{a_i^{-1}} = v_{ij},\ a_k^{r_k} = u_{kk} \rangle$$

$1 \leq i < j \leq n,\ k \in I \subset \{1,\ldots,n\},\ r_i \in \mathbb{N},\ i \in I$

w_{ij}, v_{ij}, u_{jj} words in generators $a_{j+1}, \ldots, a_n$

By induction:

$$\forall\ w \in P;\ \mathbf{w = a_1^{e_1} \ldots a_n^{e_n}}\ \text{(normal form)}$$

$$e_i \in \mathbb{Z}\ \ \&\ \ 0 \leq e_i \leq r_i\ \ \text{if}\ \ i \in I$$

$\forall$ polycyclic group has a consistent polycyclic presentation!

PC presentation is **consistent** if:

$\forall$ element in the group defined by the presentation can be represented uniquely by a word of the form

$$\mathbf{a_1^{e1} \ldots a_n^{en}} \text{ (Normal Form)}$$

$$(e_i \in Z,\ 0 < e_i < r_i,\ \ i \in I)$$

‘basis for computations w/ polycyclic groups’

PCP(P)>> CP(P) >> WP(P)

Word Problem in Polycyclic Groups
Solved effectively using collection algorithm.
Implemented in GAP and MAGMA (practical).

Conjugacy Problem in Polycyclic Groups
Algorithm by Eick-Ostheimer.
Implemented in GAP and KANT (not efficient)

Search Conjugacy Problem
in any subgroup of the General Linear group is solvable
is conjectured to be exponential.

Power Conjugacy Problem in Polycyclic Groups
is an open question!

WP in Polycyclic groups collection algorithm

WP in consistent PC presentation solved “effectively” using collection algorithm.
Computes: unique normal word for an arbitrary word in the generators.

basic idea: applies iteratedly the power and conjugate relations of the given presentation to subwords of a given word modifies the given word.

The nature of the relations asserts that an iteration of this process will eventually produce a normal word.

Complexity of Collection Algorithm

Complexity depend on the growth of exponents e_j of generators g_{i_j} occurring in intermediate stages of the algorithm while processing the word $\mathbf{g_{i_1}^{e_1} \ldots g_{i_l}^{e_l}}$

Growth can be bounded above if the considered group is finite.
In infinite PC groups: risk of an integer explosion inherent in the collection algorithm.

In praxis, collection is known as an effective method to solve WP in consistent polycyclic presentations.

Implemented in GAP and MAGMA and it has proved to be practical for finite and infinite polycyclic groups.

Collect Word

CollectWord($\mathcal{P}, w$)
  while w is not collected do
    choose an uncollected basic subword v in w
    modify v in w using the relevant basic transformation
  end while

Example of Collection Algorithm

$G = \langle g_1, \ldots, g_4 \mid g_1^3 = g_4,\ g_2^{g_1} = g_2^{-1},$
$g_3^{g_1} = g_3^{-1},\ g_3^{g_2} = g_3\, g_4^2,\ g_3^{g_2^{-1}} = g_3\, g_4^{-2} \rangle$

where all the omitted conjugate relations are trivial. We collect the word $g_2\, g_1^3$.

$g_2\, g_1^3 = g_2\, g_4$ by applying the power relation of g_1.

Multiple Conjugacy Problem

AKE can be broken if an effective algorithm to determine a multiple conjugating element a with $r_i^a = s_i$ $(1 \leq I \leq m)$ can be found (knowing that such an element exists)

Multiple CP reduces to the single CP.

Single Conjugacy Problem

Non-commutative DH and AKE can be broken if an effective algorithm to determine a conjugating element $r^a = s$ for given **r, s** $\in$ **G** can be found.

For AKE additional computation of $C_G(s)$ is necessary.
It has been proved there exists an algorithm to compute such a conjugating element and such a centralizer in a polycyclic group.

Implemented in GAP & KANT.

Algorithm for single CP

The algorithm uses induction down along a normal series with elementary or free abelian factors of the considered group G. The steps corresponding to finite factors in the series can be solved by an application of an orbit-stabilizer algorithm as typical for finite groups.

The steps corresponding to infinite factors require:

[a)] Methods from representation theory such as the computation of submodule series.

[b)] Methods from algebraic number theory such as the computation of unit groups.

[c)] Methods from finite polycyclic groups such as the finite orbit-stabilizer algorithm.

Growth Rate of Polycyclic Groups

A large growth rate would imply a large key space for the set of all possible keys, thus making the brute force search of this space intractable.

Ideally, we would like to use groups which exhibit provably exponential growth.

A large class of polycyclic groups are known to have an exponential growth rate (Wolf and Milnor).

Experiment
Running time

h(G)	WP	CP
2	0.00 sec	9.96 sec
2	0.00 sec	9.37 sec
6	0.01 sec	10.16 sec
14	0.05 sec	> **100 hrs**

Example of Polycyclic Groups

$K = \mathbb{Q}[x]/(f)$: algebraic number field for a cyclotomic polynomial

f_w (w is a primitive r-th root of unity) deg $(f_w) = \phi(n)$.

Maximal order O of K is a ring whose additive group is isomorphic to $\mathbb{Z}^n$.

Unit group U of K: f.g. abelian group

Natural split extension $G(w) = O \rtimes U$

Metabelian polycyclic group.

Some Known Complexity Result

The complexity of word problem in Nilpotent groups is in Polynomial time, indeed it is almost linear. (Holt, Gersten, Riley)

Word Problem is in NP for any finitely generated metabelian groups. (Arzhantseva, Osin)