有限アーベル群の構造計算アルゴリズムについて
A survey of the baby step giant step algorithm for the structure of a finite abelian group

松尾 和人
(情報セキュリティ大学院大学)

Kazuto Matsuo, July 18, 2006 @ Metro U.

References

- Cohen, Algorithm 5.4.1 in *A Course in Computational Algebraic Number Theory 4th ed.*, GTM 138, 2000.

- Magma, `AbelianGroup(J) : JacHyp -> GrpAb, Map`

- Teske, *A space efficient algorithm for group structure computation*, Math. Comp., 67 (1998), 1637-1663.

- M., Chao, and Tsujii, *Determination of endomorphism type of Jacobian varieties of hyperelliptic curves over finite fields*, IEICE Trans. J85-A (2002), no. 6, 677-690. in Japanese

Problem

Given:

G: finite abelian group

Find:

$r \in \mathbb{Z}_{\geq 1}$, $(d_1, d_2, \ldots, d_r) \in (\mathbb{Z}_{\geq 2})^r$ s.t.

$G \cong \bigoplus_{0<i\leq r} \mathbb{Z}/d_i\mathbb{Z}$,

$d_i \mid d_{i+1}$ for $i = 1, \ldots, r-1$

Requirements for G

- Known:
  - $\#G$
  - $\{(l_i, e_i)\}$, where $\#G = \prod_{0<i\leq s} q_i$, $q_i := l_i^{e_i}$, $\{l_i\}$: s distinct primes
- $^\forall(\alpha, \beta) \in G^2$, efficiently computable:
  - $\alpha + \beta$
  - $\alpha \stackrel{?}{=} \beta$
  - $\alpha \stackrel{?}{<} \beta$ (!)

    e.g. $G = \mathcal{J}_C(\mathbb{F}_p)$ of genus 2 HEC, $^\forall\mathcal{D} \in G$,

    $$\mathcal{D} = \begin{cases} (X^2 + u_1X + u_0, v_1X + v_0) & \\ (X + u_0, v_0) & \text{for } u_0, u_1, v_0, v_1 \in \mathbb{F}_p \\ (1, 0) & \end{cases}$$

    $u_0, u_1, v_0, v_1 \in [0, p-1] \subset \mathbb{Z}$

    $$key(\mathcal{D}) := \begin{cases} u_0 + u_1p + v_0p^2 + v_1p^3 & \\ u_0 + v_0p^2 & \text{resp.} \\ 0 & \end{cases}$$

    $\mathcal{D}_1 < \mathcal{D}_2 \Leftrightarrow key(\mathcal{D}_1) < key(\mathcal{D}_2)$

Requirements for the computer system

- A data structure s.t.

  1. Both of "Search," and "Insert" can be done within $O(\log n)$
     for n-element set
     e.g. Red-Black Tree (cf. Knuth ACP 3)
     NB: Both "Array" and "List" does not sutisfy the requirement.

  2. Each elements can be contained some informations other than its key

  – "Indexed Set" ({`@...@`}) on Magma
  – Class "set" in STL of ISO C++

Final Part

$\#G = \prod_{0<i\leq s} q_i$, $q_i := l_i^{e_i}$

$G \cong \bigoplus_{0<i\leq s} G[q_i]$,

$G[q_i] \cong \bigoplus_{0<j\leq t_i} \mathbb{Z}/l_i^{e_{(i,j)}}\mathbb{Z}$, for some t_i with

$\sum_{0<j\leq t_i} e_{(i,j)} = e_i$, $e_{(i,j)} \mid e_{(i,j+1)}$ for $0 < j < t_i$

$\Rightarrow$

$d_r = \prod_{0<i\leq s} l_i^{e_{(i,t_i)}}$, $d_{r-1} = \prod_{0<i\leq s} l_i^{e_{(i,t_i-1)}}$, $\ldots$

with $e_{(i,j)} = 0$ for $j \leq 0$ sutisfies

$G \cong \bigoplus_{0<i\leq r} \mathbb{Z}/d_i\mathbb{Z}$, $d_i \mid d_{i+1}$ for $i = 1, \ldots, r-1$

$\Rightarrow$

If one finds $e_{(i,j)}$s then the task is done.

Reduced Problem

Given: G: finite abelian group, $q = l^e$, l: prime, $l^e \mid \#G$, $l^{e+1} \nmid \#G$

Find: $r \in \mathbb{Z}_{\geq 1}$, $(d_1, d_2, \ldots, d_r) \in (\mathbb{Z}_{\geq 2})^r$ s.t.

$G[q] \cong \bigoplus_{0<i\leq r} \mathbb{Z}/d_i\mathbb{Z}$,

$d_i \mid d_{i+1}$ for $i = 1, \ldots, r-1$

$\Rightarrow$

$d_i = l^{e_i}$ (with redifinition of e_i)

$e_i \leq e_{i+1}$ for $i = 1, \ldots, r-1$

NB:

$$\begin{array}{rcl} G & \rightarrow & G[q] \\ a & \mapsto & [\#G/q]a \end{array}$$

Linear Algebra Part

If one can find

$b = {}^t(g_1, g_2, \ldots, g_k) \in G[q]^k$ s.t. $G[q] = \langle g_1, g_2, \ldots, g_k \rangle$,

and $A \in \mathbb{Z}^{k\times k}$ s.t. lower triangluar, $\det A \neq 0$, $Ab = 0$,

then

$(d_1, \ldots, d_r)$ can be obtained via. the Smith normal form of A

i.e.

${}^{\exists!}S(A) = (s_{ij}) \in \mathbb{Z}^{k\times k}$, s.t. $s_{ij} = 0$ for $i \neq j$, $s_{11} \mid s_{22} \mid \cdots \mid s_{kk}$,
and $S(A) = VAU$ with U, $V \in \mathrm{GL}(k, \mathbb{Z})$

$\Rightarrow S(A)\tilde{b} = 0$, where $\tilde{b} := U^{-1}b \Rightarrow$ diag. elts. of $S(A) = d_i$s (+1s)

NB: $\det A = \prod_{0<i\leq k} \mathrm{diag.\ elt.} = q \Rightarrow$ diag. elt. of $A =$ a power of l

Main problem from an algorithmic viewpoint

Find:

$b = {}^t(g_1, g_2, \ldots, g_k) \in G[q]^k$ s.t. $G[q] = \langle g_1, g_2, \ldots, g_k \rangle$,

and $A \in \mathbb{Z}^{k \times k}$ s.t. lower triangluar, $\det A \neq 0$, $Ab = 0$

$\Leftarrow$

Finding by a "sequential" manner

$$\begin{tikzcd}
g_1 \arrow[r] & A_1 \arrow[dl] \\
g_2 \arrow[r] & A_2 \arrow[dl] \\
{} &
\end{tikzcd}$$

Two algorithms for the main problem

- Exhaustive search
- Baby step giant step

NB: The problem can be efficiently solved by Teske's method also.

Exhaustive Search

Procedures for g_1 and g_2

$g_1 \in G[q] \setminus \{0\}$

Find minimum e_1 s.t. $0 < e_1 \le e$, $[l^{e_1}]g_1 = 0$

$A = \begin{pmatrix} l^{e_1} \end{pmatrix}$

$T = (([i]g_1, i))$ for $i = 0, \ldots, l^{e_1} - 1$

$g_2 \in G[q] \setminus T$

Find minimum e_2 s.t. $0 < e_2 \le e - e_1$, $[l^{e_2}]g_2 \in T$

If $[l^{e_2}]g_2 = [i]g_1$,

$A = \begin{pmatrix} l^{e_1} & 0 \\ -i & l^{e_2} \end{pmatrix}$

$T = (([i]g_1 + [j]g_2, (i, j)))$ for $i = 0, \ldots, l^{e_1} - 1$, $j = 0, \ldots, l^{e_2} - 1$

Procedure for g_k and the stop criterion

$g_k \in G[q] \setminus T$

Find minimum e_k s.t. $0 < e_k \leq e - (e_1 + e_2 + \cdots + e_{k-1})$, $[l^{e_k}]g_k \in T$

If $[l^{e_k}]g_k = [i_1]g_1 + [i_2]g_2 + \cdots + [i_{k-1}]g_{k-1}$,

$$A = \begin{pmatrix} l^{e_1} & 0 & \cdots & 0 \\ \vdots & l^{e_2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ -i_1 & \cdots & -i_{k-1} & l^{e_k} \end{pmatrix}$$

$T = ([i_1]g_1 + [i_2]g_2 + \cdots + [i_k]g_k, (i_1, i_2, \ldots, i_k))$ for $i_j = 0, \ldots, l^{e_j} - 1$

Stop Criterion:

If $\sum_{1 \leq i \leq k} e_i = e$ then terminate.

Complexity of main algo.

$k \leq e$, $e = O(\log q)$

- $g_i \in G[q] \setminus T$: $O\tilde{}(1)$
  - Prob. that $g \notin T \geq 1/2$
    Worst: $l = 2$, $\#T = l^{e-1} \Rightarrow$ Prob. $= (l^e - l^{e-1})/l^e = 1 - 1/2 = 1/2$
  - $g \in G[q]$: $O(k \log \#G)$ G-ops.
  - Testing $g \in T$: $O(k \log q)$ $|key|$ bit-ops. by "Search"
- Finding minimum e_i s.t. $0 < e_i \leq e - (e_1 + e_2 + \cdots + e_{i-1})$, $[l^{e_i}]g_i \in T$, and $(i_1, \ldots, i_{i-1})$ s.t. $[l^{e_i}]g_k = [i_1]g_1 + [i_2]g_2 + \cdots + [i_{i-1}]g_{i-1}$: $O\tilde{}(1)$
  - $[l^{e_j}]g_i \in G[q]$: $O(ke \log l)$ G-ops.
  - Testing $[l^{e_i}]g_i \in T$: $O(ke \log q)$ $|key|$ bit-ops.
- Updating A: $O\tilde{}(1)$

◦ Updating T: $O\tilde{}(q)$ G-ops.

$\Rightarrow O\tilde{}(q)$ G-ops.

Total Complexity

- Main Part: $O\tilde{}(q)$ G-ops., $q = \max(q_i)$

- Linear Algebra Part: $O\tilde{}(1)$

- Final Part: $O\tilde{}(1)$

$\Rightarrow O\tilde{}(q)$ G-ops., $q = \max(q_i)$

and

$O(q)$ spaces

Shanks's baby step giant step algorithm

Given:

G: finite abelian group, $\alpha \in G$, $B \in \mathbb{N}$ s.t. $\#\langle\alpha\rangle < B$

Find:

$n \in \mathbb{N}$ s.t. $n < B$, $n = c\#\langle\alpha\rangle$ with some $c \in \mathbb{N}$

Idea:

m-adic representation of n

i.e.

For given $m \in \mathbb{N}$,

${}^{\exists}n_0, n_1 \in \mathbb{Z}$ s.t. $0 \leq n_0 < m$, $0 \leq n_1 < \lceil B/m \rceil$ with

$n = n_1 m + n_0$

Input: G: finite abelian group, $\alpha \in G$, $B \in \mathbb{N}$ with $\#\langle\alpha\rangle < B$, $m \leq B$
Output: $n \in \mathbb{N}$ s.t. $n < B$, $n = c\#\langle\alpha\rangle$ with some $c \in \mathbb{N}$

$S \leftarrow \{[i]\alpha \mid i = 0, \dots, m-1\}$ with their indecies `/*Baby Step*/`
$\beta = [-m]\alpha$, $\gamma = 0$, $i \leftarrow 0$
repeat `/*Giant Step*/`
    $\gamma \leftarrow \gamma + \beta$, $i \leftarrow i + 1$
until $\gamma \in S$
$n \leftarrow im + j$, if $[i]\beta = [j]\alpha$
return n

$O(m + B/m)$ G-ops. with $O(m)$ spaces

$\Rightarrow$

Setting $m \approx \sqrt{B}$ gives the best performance i.e.

$O(\sqrt{B})$ ops. with $O(\sqrt{B})$ spaces.

The Baby Step Giant Step Algorithm for the Structure of A Finite Abelian Group

Procedure for g_1

$g_1 \in G[q] \setminus \{0\}$

Find minimum e_1 s.t. $0 < e_1 \leq e$, $[l^{e_1}]g_1 = 0$

$A = \left(l^{e_1}\right)$

$T = (([i]g_1, i))$ for $i = 0, \ldots, l^{e_1} - 1$

$m = \left\lceil l^{e_1/2} \right\rceil$

$S = \{0, g_1, [2]g_1, \ldots, [m-1]g_1\}$

$L = \{0, [-m]g_1, [-2m]g_1, \ldots, [-m^2]g_1\}$

Procedure for g_2

$g_2 \in G[q] \setminus T$

$\longrightarrow g_2 \in G[q] \setminus \{0\}$

Find minimum e_2 s.t. $0 < e_2 \le e - e_1$, $[l^{e_2}]g_2 \in T$

$\longrightarrow$ Find minimum e_2, and $(\alpha, \beta) \in S \times L$ s.t. $0 \le e_2 \le e - e_1$, $[l^{e_2}]g_2 + \alpha = \beta$

If $[l^{e_2}]g_2 = [i]g_1$,

$\longrightarrow$ If $\alpha = [i]g_1$, $\beta = [j]g_1$,

$$A = \begin{pmatrix} l^{e_1} & 0 \\ -i & l^{e_2} \end{pmatrix}$$

$$\longrightarrow A = \begin{pmatrix} l^{e_1} & 0 \\ i - j & l^{e_2} \end{pmatrix}$$

$T = (([i]g_1 + [j]g_2, (i, j)))$ for $i = 0, \ldots, l^{e_1} - 1$, $j = 0, \ldots, l^{e_2} - 1$

$\longrightarrow m = \lceil l^{e_2/2} \rceil$, $S \leftarrow \bigcup_{0 \le i < m}([i]g_2 + S)$, $L \leftarrow \bigcup_{0 \le i \le m}([-im]g_2 + L)$

Procedure for g_k

$g_k \in G[q] \setminus \{0\}$

Find minimum e_k, and $(\alpha, \beta) \in S \times L$ s.t. $[l^{e_k}]g_k + \alpha = \beta$

If $\alpha = [i_1]g_1 + [i_2]g_2 + \cdots + [i_{k-1}]g_{k-1}$, $\beta = [j_1]g_1 + [j_2]g_2 + \cdots + [j_{k-1}]g_{k-1}$,

$$A = \begin{pmatrix} l^{e_1} & 0 & \cdots & 0 \\ \vdots & l^{e_2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ (i_1 - j_1) & \cdots & (i_{k-1} - j_{k-1}) & l^{e_k} \end{pmatrix}$$

$m = \lceil l^{e_k/2} \rceil$

$S \leftarrow \bigcup_{0 \le i < m} ([i]g_k + S)$

$L \leftarrow \bigcup_{0 \le i \le m} ([-im]g_k + L)$

Complexity for k

- $g_k \in G[q] \setminus \{0\}$: $O\tilde{}(1)$
- Find minimum e_k, and $(\alpha, \beta) \in S \times L$ s.t. $[l^{e_k}]g_k + \alpha = \beta$: $O\tilde{}(\#S)$
- Updating A: $O\tilde{}(1)$

◦ $S \leftarrow \bigcup_{0 \leq i < m}([i]g_k + S)$: $O\tilde{}(\lceil l^{e_k/2} \rceil \#S)$

◦ $L \leftarrow \bigcup_{0 \leq i \leq m}([-im]g_k + L)$: $O\tilde{}(\lceil l^{e_k/2} \rceil \#L)$

$\#S \approx \#L$

$\Rightarrow$

Total complexity $= \displaystyle\sum_{1 \leq i \leq k} O\tilde{}\left(\left\lceil l^{e_i/2} \right\rceil \#S\right)$

Total Complexity

$$= \sum_{1 \le i \le k} O\tilde{}\left(\left\lceil l^{e_i/2} \right\rceil \#S\right)$$

$$(\#S)_1 = \left\lceil l^{e_1/2} \right\rceil$$

$$(\#S)_2 = \left\lceil l^{e_2/2} \right\rceil (\#S)_1 = \left\lceil l^{e_1/2} \right\rceil \left\lceil l^{e_2/2} \right\rceil$$

$$\vdots$$

$$(\#S)_k = \left\lceil l^{e_k/2} \right\rceil (\#S)_{k-1} = O(\sqrt{q})$$

$$\Rightarrow$$

Total complexity $= \displaystyle\sum_{1 \le i \le k} O\tilde{}\left(\left\lceil l^{e_i/2} \right\rceil \#S\right) = O\tilde{}\,(\sqrt{q})$