Computational algebraic number theory tackles lattice-based cryptography

Daniel J. Bernstein
University of Illinois at Chicago &
Technische Universiteit Eindhoven

Moving to the left
Moving to the right
Big generator
Moving through the night
—Yes, "Big Generator", 1987

The short-generator problem

Take degree-n number field K.
i.e. field $K \subseteq \mathbf{C}$ with $\operatorname{len}_{\mathbf{Q}} K = n$.

(Weaker specification: field K with $\mathbf{Q} \subseteq K$ and $\operatorname{len}_{\mathbf{Q}} K = n$.)

e.g. $n = 2$; $K = \mathbf{Q}(i) = \mathbf{Q} \oplus \mathbf{Q}i \hookrightarrow\!\!\!\rightarrow \mathbf{Q}[x]/(x^2 + 1)$.
e.g. $n = 256$; $\zeta = \exp(\pi i/n)$; $K = \mathbf{Q}(\zeta) \hookrightarrow\!\!\!\rightarrow \mathbf{Q}[x]/(x^n + 1)$.
e.g. $n = 660$; $\zeta = \exp(2\pi i/661)$; $K = \mathbf{Q}(\zeta) \hookrightarrow\!\!\!\rightarrow \mathbf{Q}[x]/(x^n + \cdots + 1)$.
e.g. $K = \mathbf{Q}(\sqrt{2}, \sqrt{3}, \sqrt{5}, \ldots, \sqrt{29})$.

Define $\mathcal{O} = \overline{\mathbf{Z}} \cap K$; subring of K.
$\mathcal{O} \hookrightarrow\!\!\!\!\rightarrow \mathbf{Z}^n$ as $\mathbf{Z}$-modules.

Nonzero ideals of $\mathcal{O}$
factor uniquely as products of
powers of prime ideals of $\mathcal{O}$.

e.g. $K = \mathbf{Q}(i) \hookrightarrow\!\!\!\!\rightarrow \mathbf{Q}[x]/(x^2+1)$
$\Rightarrow \mathcal{O} = \mathbf{Z}[i] \hookrightarrow\!\!\!\!\rightarrow \mathbf{Z}[x]/(x^2+1)$.
e.g. $\zeta = \exp(\pi i/256)$, $K = \mathbf{Q}(\zeta)$
$\Rightarrow \mathcal{O} = \mathbf{Z}[\zeta] \hookrightarrow\!\!\!\!\rightarrow \mathbf{Z}[x]/(x^{256}+1)$.
e.g. $\zeta = \exp(2\pi i/661)$, $K = \mathbf{Q}(\zeta)$
$\Rightarrow \mathcal{O} = \mathbf{Z}[\zeta] \hookrightarrow\!\!\!\!\rightarrow \cdots$.
e.g. $K = \mathbf{Q}(\sqrt{5}) \Rightarrow \mathcal{O} =$
$\mathbf{Z}[(1+\sqrt{5})/2] \hookrightarrow\!\!\!\!\rightarrow \mathbf{Z}[x]/(x^2-x-1)$.

The short-generator problem:

Find "short" nonzero $g \in \mathcal{O}$ given the principal ideal $g\mathcal{O}$.

e.g. $\zeta = \exp(\pi i/4)$; $K = \mathbf{Q}(\zeta)$; $\mathcal{O} = \mathbf{Z}[\zeta] \hookrightarrow\!\!\!\!\rightarrow \mathbf{Z}[x]/(x^4+1)$.

The $\mathbf{Z}$-submodule of $\mathcal{O}$ gen by

$201 - 233\zeta - 430\zeta^2 - 712\zeta^3$,

$935 - 1063\zeta - 1986\zeta^2 - 3299\zeta^3$,

$979 - 1119\zeta - 2092\zeta^2 - 3470\zeta^3$,

$718 - 829\zeta - 1537\zeta^2 - 2546\zeta^3$

is an ideal I of $\mathcal{O}$.

Can you find a short $g \in \mathcal{O}$ such that $I = g\mathcal{O}$?

The lattice perspective

Use LLL to quickly find
short elements of lattice
$\mathbf{Z}A + \mathbf{Z}B + \mathbf{Z}C + \mathbf{Z}D$ where
$A = (201, -233, -430, -712)$,
$B = (935, -1063, -1986, -3299)$,
$C = (979, -1119, -2092, -3470)$,
$D = (718, -829, -1537, -2546)$.

The lattice perspective

Use LLL to quickly find short elements of lattice $\mathbf{Z}A + \mathbf{Z}B + \mathbf{Z}C + \mathbf{Z}D$ where

$A = (201, -233, -430, -712)$,
$B = (935, -1063, -1986, -3299)$,
$C = (979, -1119, -2092, -3470)$,
$D = (718, -829, -1537, -2546)$.

Find $(3, 1, 4, 1)$ as $-37A + 3B - 7C + 16D$.
This was my original g.

The lattice perspective

Use LLL to quickly find
short elements of lattice
$\mathbf{Z}A + \mathbf{Z}B + \mathbf{Z}C + \mathbf{Z}D$ where
$A = (201, -233, -430, -712)$,
$B = (935, -1063, -1986, -3299)$,
$C = (979, -1119, -2092, -3470)$,
$D = (718, -829, -1537, -2546)$.

Find $(3, 1, 4, 1)$ as
$-37A + 3B - 7C + 16D$.
This was my original g.

Also find, e.g., $(-4, -1, 3, 1)$.
Multiplying by root of unity
(here ζ^2) preserves shortness.

For much larger n:

LLL almost never finds g.
Big gap between size of g
and size of "short" vectors
that LLL typically finds in I.

For much larger n:

LLL almost never finds g.
Big gap between size of g
and size of “short” vectors
that LLL typically finds in I.

Increased BKZ block size:
reduced gap but slower.

For much larger n:

LLL almost never finds g.
Big gap between size of g
and size of “short” vectors
that LLL typically finds in I.

Increased BKZ block size:
reduced gap but slower.

Fancier lattice algorithms:
Under reasonable assumptions,
2015 Laarhoven–de Weger
finds g in time $\approx 1.23^n$.
Big progress compared to, e.g.,
2008 Nguyen–Vidick ($\approx 1.33^n$)
but still exponential time.

Exploiting factorization

Use LLL, BKZ, etc. to
generate rather short $\alpha \in g\mathcal{O}$.
What happens if $\alpha\mathcal{O} \neq g\mathcal{O}$?

Pure lattice approach: Discard α.
Work much harder, find shorter α.

Exploiting factorization

Use LLL, BKZ, etc. to
generate rather short $\alpha \in g\mathcal{O}$.
What happens if $\alpha\mathcal{O} \neq g\mathcal{O}$?

Pure lattice approach: Discard α.
Work much harder, find shorter α.

Alternative: Gain information
from factorization of ideals.

Exploiting factorization

Use LLL, BKZ, etc. to
generate rather short $\alpha \in g\mathcal{O}$.
What happens if $\alpha\mathcal{O} \neq g\mathcal{O}$?

Pure lattice approach: Discard α.
Work much harder, find shorter α.

Alternative: Gain information
from factorization of ideals.

e.g. If $\alpha_1\mathcal{O} = g\mathcal{O} \cdot P^2 \cdot Q^2$

Exploiting factorization

Use LLL, BKZ, etc. to
generate rather short $\alpha \in g\mathcal{O}$.
What happens if $\alpha\mathcal{O} \neq g\mathcal{O}$?

Pure lattice approach: Discard α.
Work much harder, find shorter α.

Alternative: Gain information
from factorization of ideals.

e.g. If $\alpha_1\mathcal{O} = g\mathcal{O} \cdot P^2 \cdot Q^2$
and $\alpha_2\mathcal{O} = g\mathcal{O} \cdot P \cdot Q^3$

Exploiting factorization

Use LLL, BKZ, etc. to generate rather short $\alpha \in g\mathcal{O}$.
What happens if $\alpha\mathcal{O} \neq g\mathcal{O}$?

Pure lattice approach: Discard α.
Work much harder, find shorter α.

Alternative: Gain information from factorization of ideals.

e.g. If $\alpha_1\mathcal{O} = g\mathcal{O} \cdot P^2 \cdot Q^2$
and $\alpha_2\mathcal{O} = g\mathcal{O} \cdot P \cdot Q^3$
and $\alpha_3\mathcal{O} = g\mathcal{O} \cdot P \cdot Q^2$

Exploiting factorization

Use LLL, BKZ, etc. to
generate rather short $\alpha \in g\mathcal{O}$.
What happens if $\alpha\mathcal{O} \neq g\mathcal{O}$?

Pure lattice approach: Discard α.
Work much harder, find shorter α.

Alternative: Gain information
from factorization of ideals.

e.g. If $\alpha_1\mathcal{O} = g\mathcal{O} \cdot P^2 \cdot Q^2$
and $\alpha_2\mathcal{O} = g\mathcal{O} \cdot P \cdot Q^3$
and $\alpha_3\mathcal{O} = g\mathcal{O} \cdot P \cdot Q^2$ then
$P = \alpha_1\alpha_3^{-1}\mathcal{O}$ and $Q = \alpha_2\alpha_3^{-1}\mathcal{O}$
and $g\mathcal{O} = \alpha_1^{-1}\alpha_2^{-2}\alpha_3^4\mathcal{O}$.

General strategy: For many α's, factor $\alpha\mathcal{O}$ into products of powers of some primes and $g\mathcal{O}$.

Solve system of equations
to find generator for $g\mathcal{O}$
as product of powers of the α's.

General strategy: For many α's, factor $\alpha\mathcal{O}$ into products of powers of some primes and $g\mathcal{O}$.

Solve system of equations
to find generator for $g\mathcal{O}$
as product of powers of the α's.

"Can the system be solved?"

— Becomes increasingly reasonable to expect as the number of equations approaches and passes the number of primes.

General strategy: For many α's, factor $\alpha\mathcal{O}$ into products of powers of some primes and $g\mathcal{O}$.

Solve system of equations to find generator for $g\mathcal{O}$ as product of powers of the α's.

"Can the system be solved?"

— Becomes increasingly reasonable to expect as the number of equations approaches and passes the number of primes.

"But {primes} is infinite!"

— Restrict to a “factor base”:
e.g., all primes of norm $\leq y$.

— Restrict to a “factor base”: e.g., all primes of norm $\le y$.

“But what if $\alpha\mathcal{O}$ doesn't factor into those primes?”

— Restrict to a "factor base": e.g., all primes of norm $\le y$.

"But what if $\alpha\mathcal{O}$ doesn't factor into those primes?"

— Then throw it away. But often it *does* factor.

— Restrict to a “factor base”: e.g., all primes of norm $\le y$.

“But what if $\alpha\mathcal{O}$ doesn't factor into those primes?”

— Then throw it away. But often it *does* factor.

Familiar issue from “index calculus” DL methods, CFRAC, LS, QS, NFS, etc. Model the norm of $(\alpha/g)\mathcal{O}$ as “random” integer in $[1, x]$; y-smoothness chance $\approx 1/y$ if $\log y \approx \sqrt{(1/2)\log x \log\log x}$.

Variation: Ignore $g\mathcal{O}$.

Generate rather short $\alpha \in \mathcal{O}$,

factor $\alpha\mathcal{O}$ into small primes.

After enough α's,

solve system of equations;

obtain generator for each prime.

Variation: Ignore $g\mathcal{O}$.

Generate rather short $\alpha \in \mathcal{O}$,

factor $\alpha\mathcal{O}$ into small primes.

After enough α's,

solve system of equations;

obtain generator for each prime.

After this precomputation,

factor *one* $\alpha\mathcal{O} \subseteq g\mathcal{O}$;

obtain generator for $g\mathcal{O}$.

Variation: Ignore $g\mathcal{O}$.
Generate rather short $\alpha \in \mathcal{O}$,
factor $\alpha\mathcal{O}$ into small primes.
After enough α's,
solve system of equations;
obtain generator for each prime.
After this precomputation,
factor *one* $\alpha\mathcal{O} \subseteq g\mathcal{O}$;
obtain generator for $g\mathcal{O}$.

"Do all primes have generators?"

Variation: Ignore $g\mathcal{O}$.
Generate rather short $\alpha \in \mathcal{O}$,
factor $\alpha\mathcal{O}$ into small primes.
After enough α's,
solve system of equations;
obtain generator for each prime.
After this precomputation,
factor *one* $\alpha\mathcal{O} \subseteq g\mathcal{O}$;
obtain generator for $g\mathcal{O}$.

"Do all primes have generators?"

— Standard heuristics:
For many (most?) number fields,
yes; but for big cyclotomics, no!
Modulo a few small primes, yes.

$\{$principal nonzero ideals$\}$ is kernel of a semigroup map $\{$nonzero ideals$\} \twoheadrightarrow C$ where C is a finite abelian group, the "class group of K".

Fundamental object of study in algebraic number theory.

$\{$principal nonzero ideals$\}$ is kernel of a semigroup map $\{$nonzero ideals$\} \twoheadrightarrow C$ where C is a finite abelian group, the "class group of K".

Fundamental object of study in algebraic number theory.

Factoring many small $\alpha\mathcal{O}$ is a standard textbook method of computing class group and generators of ideals.

{principal nonzero ideals} is kernel of a semigroup map {nonzero ideals} $\twoheadrightarrow C$ where C is a finite abelian group, the "class group of K".

Fundamental object of study in algebraic number theory.

Factoring many small $\alpha\mathcal{O}$ is a standard textbook method of computing class group and generators of ideals.

Also compute unit group $\mathcal{O}^*$ via ratios of generators.

Big generator

Smart–Vercauteren: "However this method is likely to produce a generator of large height, i.e., with large coefficients. Indeed so large, that writing the obtained generator down as a polynomial in θ may take exponential time."

Indeed, generator found for $g\mathcal{O}$ is product of powers of various α's. Must be gu for some $u \in \mathcal{O}^*$, but extremely unlikely to be g.

Big generator

Smart–Vercauteren: “However this method is likely to produce a generator of large height, i.e., with large coefficients. Indeed so large, that writing the obtained generator down as a polynomial in θ may take exponential time.”

Indeed, generator found for $g\mathcal{O}$ is product of powers of various α’s. Must be gu for some $u \in \mathcal{O}^*$, but extremely unlikely to be g.

How do we find g from gu?

There are exactly n distinct ring maps $\varphi_1, \ldots, \varphi_n : K \to \mathbf{C}$.

There are exactly n distinct ring maps $\varphi_1, \ldots, \varphi_n : K \to \mathbf{C}$.

Define $\mathrm{Log} : K^* \to \mathbf{R}^n$ by $\mathrm{Log} = (\log|\varphi_1|, \ldots, \log|\varphi_n|)$.

There are exactly n distinct ring maps $\varphi_1, \ldots, \varphi_n : K \to \mathbf{C}$.

Define $\operatorname{Log} : K^* \to \mathbf{R}^n$ by $\operatorname{Log} = (\log|\varphi_1|, \ldots, \log|\varphi_n|)$.

$\operatorname{Log} \mathcal{O}^*$ is a lattice of rank $r_1 + r_2 - 1$ where

$$r_1 = \#\{i : \varphi_i(K) \subseteq \mathbf{R}\},$$
$$2r_2 = \#\{i : \varphi_i(K) \not\subseteq \mathbf{R}\}.$$

There are exactly n distinct ring maps $\varphi_1, \ldots, \varphi_n : K \to \mathbf{C}$.

Define $\mathrm{Log} : K^* \to \mathbf{R}^n$ by $\mathrm{Log} = (\log|\varphi_1|, \ldots, \log|\varphi_n|)$.

$\mathrm{Log}\,\mathcal{O}^*$ is a lattice of rank $r_1 + r_2 - 1$ where

$$r_1 = \#\{i : \varphi_i(K) \subseteq \mathbf{R}\},$$
$$2r_2 = \#\{i : \varphi_i(K) \not\subseteq \mathbf{R}\}.$$

e.g. $\zeta = \exp(\pi i/256)$, $K = \mathbf{Q}(\zeta)$: images of ζ under ring maps are $\zeta, \zeta^3, \zeta^5, \ldots, \zeta^{511}$. $r_1 = 0$; $r_2 = 128$; rank 127.

Compute $\operatorname{Log} g u$
as sum of multiples of $\operatorname{Log} \alpha$
for the original α's.

Compute $\operatorname{Log} g u$
as sum of multiples of $\operatorname{Log} \alpha$
for the original α's.

Find elements of $\operatorname{Log} \mathcal{O}^*$
close to $\operatorname{Log} g u$.

Compute $\operatorname{Log} g u$
as sum of multiples of $\operatorname{Log} \alpha$
for the original α's.

Find elements of $\operatorname{Log} \mathcal{O}^*$
close to $\operatorname{Log} g u$.

This is a close-vector problem
("bounded-distance decoding").
"Embedding" heuristic:
CVP as fast as SVP.

Compute $\operatorname{Log} gu$
as sum of multiples of $\operatorname{Log} \alpha$
for the original α's.

Find elements of $\operatorname{Log} \mathcal{O}^*$
close to $\operatorname{Log} gu$.

This is a close-vector problem
("bounded-distance decoding").
"Embedding" heuristic:
CVP as fast as SVP.

This finds $\operatorname{Log} u$.
Easily reconstruct g
up to a root of unity.
$\#\{\text{roots of unity}\}$ is small.

A subfield-logarithm attack

(2014.02 Bernstein)

Say we know $\operatorname{Log}\operatorname{norm}_{K:F} g$ for a proper subfield $F \subset K$.

A subfield-logarithm attack

(2014.02 Bernstein)

Say we know $\operatorname{Log}\operatorname{norm}_{K:F} g$
for a proper subfield $F \subset K$.

We also know $\operatorname{Log}\operatorname{norm}_{K:F} gu$,
so we know $\operatorname{Log}\operatorname{norm}_{K:F} u$.

A subfield-logarithm attack

(2014.02 Bernstein)

Say we know $\operatorname{Log}\operatorname{norm}_{K:F} g$
for a proper subfield $F \subset K$.

We also know $\operatorname{Log}\operatorname{norm}_{K:F} gu$,
so we know $\operatorname{Log}\operatorname{norm}_{K:F} u$.

This linearly constrains $\operatorname{Log} u$
to a shifted sublattice of $\operatorname{Log}\mathcal{O}^*$.
Number of independent
constraints: unit rank for F.

A subfield-logarithm attack

(2014.02 Bernstein)

Say we know $\operatorname{Log}\operatorname{norm}_{K:F} g$
for a proper subfield $F \subset K$.

We also know $\operatorname{Log}\operatorname{norm}_{K:F} gu$,
so we know $\operatorname{Log}\operatorname{norm}_{K:F} u$.

This linearly constrains $\operatorname{Log} u$
to a shifted sublattice of $\operatorname{Log}\mathcal{O}^*$.
Number of independent
constraints: unit rank for F.

Find elements close to $\operatorname{Log} gu$.
Lower-dimension lattice problem,
if unit rank of F is positive.

Start by recursively computing $\operatorname{Log}\operatorname{norm}_{K:F} g$ via norm of $g\mathcal{O}$ for *each* $F \subset K$.

Various constraints on $\operatorname{Log} u$, depending on subfield structure.

Start by recursively computing $\operatorname{Log}\operatorname{norm}_{K:F} g$ via norm of $g\mathcal{O}$ for *each* $F \subset K$.

Various constraints on $\operatorname{Log} u$, depending on subfield structure.

e.g. $\zeta = \exp(2\pi i/661)$, $K = \mathbf{Q}(\zeta)$. Degrees of subfields of K:

660

330 220 132 60

165 110 66 44 30 20 12

55 33 22 15 10 6 4

11 5 3 2

1

Most extreme case:

Composite of quadratics, such as $K = \mathbf{Q}(\sqrt{2}, \sqrt{3}, \sqrt{5}, \ldots, \sqrt{29})$.

CVP becomes trivial!

Most extreme case:

Composite of quadratics, such as $K = \mathbf{Q}(\sqrt{2}, \sqrt{3}, \sqrt{5}, \ldots, \sqrt{29})$.

CVP becomes trivial!

Opposite extreme: prime degree; the only proper subfield is $\mathbf{Q}$.

My recommendation: big Galois group; e.g., $\mathbf{Q}[x]/(x^p - x - 1)$.

Many intermediate cases.

Most extreme case:
Composite of quadratics, such as
$K = \mathbf{Q}(\sqrt{2}, \sqrt{3}, \sqrt{5}, \ldots, \sqrt{29})$.
CVP becomes trivial!

Opposite extreme: prime degree;
the only proper subfield is $\mathbf{Q}$.
My recommendation: big Galois
group; e.g., $\mathbf{Q}[x]/(x^p - x - 1)$.

Many intermediate cases.

Confused summary by Cramer–Ducas–Peikert–Regev: method "may yield slightly subexponential runtimes in *cyclotomic* rings of *highly smooth* index".

Further improvements: 1, 2

1 2014.10 Campbell–Groves–Shepherd: Quickly solve CVP for **cyclotomics** using known (good) basis for cyclotomic units.

Further improvements: 1, 2

1 2014.10 Campbell–Groves–Shepherd: Quickly solve CVP for **cyclotomics** using known (good) basis for cyclotomic units.

Analysis in paper is bogus, but algorithm is very fast.

Further improvements: 1, 2

1 2014.10 Campbell–Groves–Shepherd: Quickly solve CVP for **cyclotomics** using known (good) basis for cyclotomic units.

Analysis in paper is bogus, but algorithm is very fast.

Plagiarized and properly analyzed by Cramer–Ducas–Peikert–Regev.

Further improvements: 1, 2

1 2014.10 Campbell–Groves–Shepherd: Quickly solve CVP for **cyclotomics** using known (good) basis for cyclotomic units.

Analysis in paper is bogus, but algorithm is very fast.

Plagiarized and properly analyzed by Cramer–Ducas–Peikert–Regev.

2 2015.01 Song announcement: Fast **quantum** algorithm for gu. "PIP . . . solved [BiasseSong'14]". But paper not available yet.