Ideal-SVP is Hard for Small-Norm Uniform Prime Ideals

Joël Felderhoff, Alice Pellet-Mary, Damien Stehlé and Benjamin Wesolowski

INRIA Lyon, ENS de Lyon

Contributions

- New reduction: $\mathcal{P}^{-1}$-ideal-SVP to $\mathcal{P}$-ideal-SVP.
- Application: new distribution of NTRU instances with difficulty based on wc-ideal-SVP.

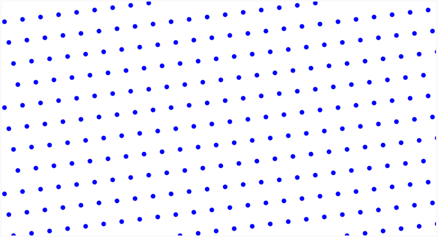
Contributions

- New reduction: $\mathcal{P}^{-1}$-$\mathrm{ideal\text{-}SVP}$ to $\mathcal{P}$-$\mathrm{ideal\text{-}SVP}$.
- Application: new distribution of NTRU instances with difficulty based on $\mathrm{wc\text{-}ideal\text{-}SVP}$.

To appear in the proceedings of **TCC 2023**. Available on:
`https://eprint.iacr.org/2023/1370`

Definitions

Lattices



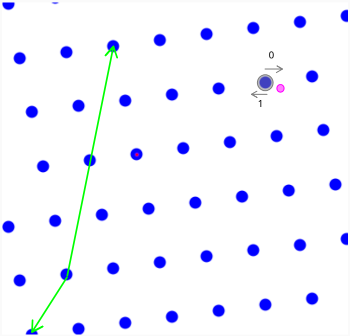
A 2-dimensional lattice

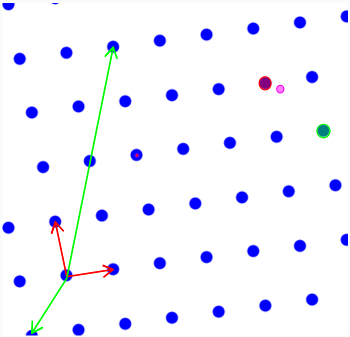
Definition

For $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{Z}^n$ linearly independent, the lattice spanned by the basis $\mathbf{b}_1, \ldots, \mathbf{b}_n$ is $\mathcal{L} = \sum_i \mathbb{Z} \cdot \mathbf{b}_i \subset \mathbb{R}^n$.
It is discrete and has a shortest non-zero vector.

Finding any short non-zero vector in $\mathcal{L}$ given the $(\mathbf{b}_i)_i$ is hard in general.

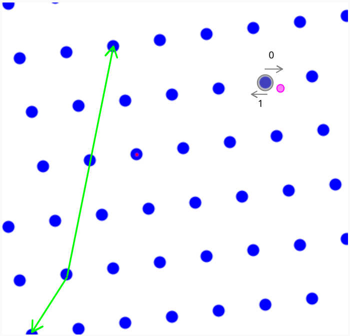
Lattice-based cryptography

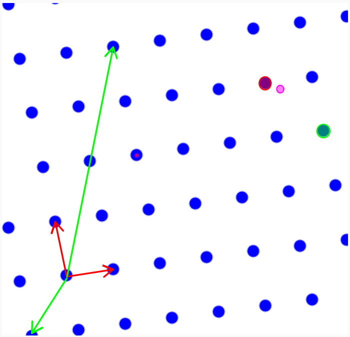




Lattice cryptography (toy example). B_{pk} and B_{sk} are the basis of a lattice $\mathcal{L}$.

Lattice-based cryptography

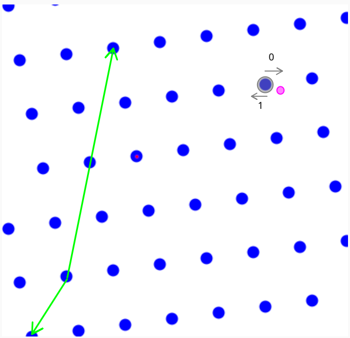


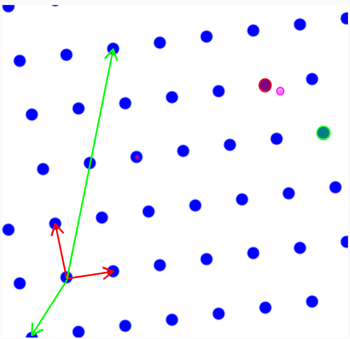


Lattice cryptography (toy example). B_{pk} and B_{sk} are the basis of a lattice $\mathcal{L}$.

Secure as long as it is hard to find B_{sk} given B_{pk}.
This problem is the Shortest-Vector-Problem (SVP).

Lattice-based cryptography





Lattice cryptography (toy example). B_{pk} and B_{sk} are the basis of a lattice $\mathcal{L}$.

Secure as long as it is hard to find B_{sk} given B_{pk}. This problem is the Shortest-Vector-Problem (SVP).

Note: $\mathcal{L}$ must be chosen at random.

Number fields and Ideals

We use the field $K = \mathbb{Q}[X]/(X^n + 1)$, $\mathcal{O}_K = \mathbb{Z}[X]/(X^n + 1)$ for $n = 2^r$.
(K a number field, $\mathcal{O}_K$ its ring of integers).

The size of an element $a \in K$ is $\|a\| = \left(\sum_i |a_i|^2\right)^{1/2}$.
The size of an element is the ℓ_2-norm of its Minkowski embedding.

Number fields and Ideals

We use the field $K = \mathbb{Q}[X]/(X^n + 1)$, $\mathcal{O}_K = \mathbb{Z}[X]/(X^n + 1)$ for $n = 2^r$.
(K a number field, $\mathcal{O}_K$ its ring of integers).

The size of an element $a \in K$ is $\|a\| = \left(\sum_i |a_i|^2\right)^{1/2}$.

The size of an element is the ℓ_2-norm of its Minkowski embedding.

Definition (Ideal)

A set $\mathfrak{a} \subseteq K$ is an ideal if it is discrete, stable by addition and by multiplication by any element of $\mathcal{O}_K$. It is then a lattice.

Norm of an ideal: $\mathcal{N}(I) = \mathrm{Vol}(I)/\sqrt{\Delta_K} \in \mathbb{Z}$.

Ideal inverse and factorization

Let $\mathfrak{a}, \mathfrak{b}$ ideals of K, and $a \in K$.

Principal ideal

$(a) = \{x \cdot a, x \in \mathcal{O}_K\}$.

Multiplication and inverse

$\mathfrak{a} \cdot \mathfrak{b} = \left\{\sum_i a_i \cdot b_i\right\}, \mathfrak{a}^{-1} = \{x \in K, x \cdot \mathfrak{a} \subseteq \mathcal{O}_K\}$.
We have that $\mathfrak{a} \cdot \mathfrak{a}^{-1} = \mathcal{O}_K$.

Factorization

There exists a set of prime ideals $\mathcal{P}$ such that any $\mathfrak{a} \subset K$ can be written in a unique way

$$\mathfrak{a} = \prod_{\mathfrak{p} \in \mathcal{P}} \mathfrak{p}^{\nu_{\mathfrak{p}}(\mathfrak{a})}.$$

The problem ideal-HSVP

Definition (ideal-HSVP$_\gamma$)

Given an ideal $\mathfrak{a} \subseteq K$, find $x \in \mathfrak{a} \setminus \{0\}$ with $\|x\| \leq \gamma \cdot \mathrm{Vol}(\mathfrak{a})^{1/d}$.

Ideal lattices are **not typical lattices**. E.g., they verify $\lambda_1(I) \approx \lambda_d(I)$.

[1] [CDPR16, CDW17, PHS19]

The problem $\mathrm{ideal\text{-}HSVP}$

Definition ($\mathrm{ideal\text{-}HSVP}_\gamma$)

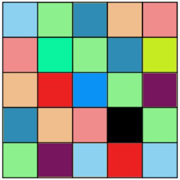
Given an ideal $\mathfrak{a} \subseteq K$, find $x \in \mathfrak{a} \setminus \{0\}$ with $\|x\| \leq \gamma \cdot \mathrm{Vol}(\mathfrak{a})^{1/d}$.

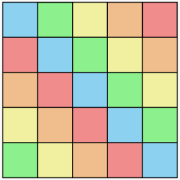
Ideal lattices are **not typical lattices**. E.g., they verify $\lambda_1(I) \approx \lambda_d(I)$.

- There are specifics attacks on ideal lattices[1].
- Ideals are the simplest examples of module lattices (they are rank-1 modules), used in real world applications (KYBER, DILITHIUM).
- $\mathrm{ideal\text{-}HSVP}$ is related to other structured lattice problems (Module-SVP, NTRU, RingLWE).

[1] [CDPR16, CDW17, PHS19]

Why small ideal lattices?

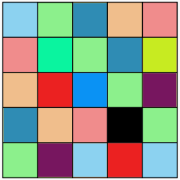


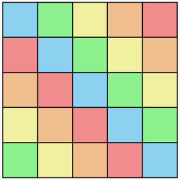


Typical lattice basis: $O(d^2)$ integers vs **Ideal lattice basis**: $O(d)$ integers.[2]

[2] Images from [Qua14]

Why small ideal lattices?



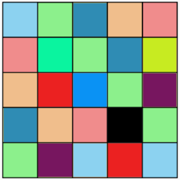


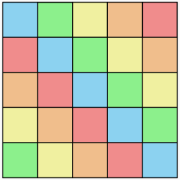
Typical lattice basis: $O(d^2)$ integers vs **Ideal lattice basis**: $O(d)$ integers.[2]

Bitsize of a typical element of $\mathfrak{a}$ is $\log(\mathcal{N}(\mathfrak{a}))$.
$\rightarrow$ We want $\mathcal{N}(\mathfrak{a}) \approx \mathrm{poly}(d)^d$ in order to have small keys.

[2] Images from [Qua14]

Why small ideal lattices?





Typical lattice basis: $O(d^2)$ integers vs **Ideal lattice basis**: $O(d)$ integers.[2]

Bitsize of a typical element of $\mathfrak{a}$ is $\log(\mathcal{N}(\mathfrak{a}))$.
$\rightarrow$ We want $\mathcal{N}(\mathfrak{a}) \approx \mathrm{poly}(d)^d$ in order to have small keys.

Also: faster algorithms.

[2] Images from [Qua14]

Worst-case to Average-case reduction

Worst-case: Solve $\mathcal{P}$ for *all* instance of $\mathcal{P}$ (for the worst instance).

Average-case for D: Solve $\mathcal{P}$ for $I \leftarrow D$ *with non-negligible probability*.

Average-case: "Find the secret key given a random public key".

Worst-case to Average-case reduction

Worst-case: Solve $\mathcal{P}$ for *all* instance of $\mathcal{P}$ (for the worst instance).

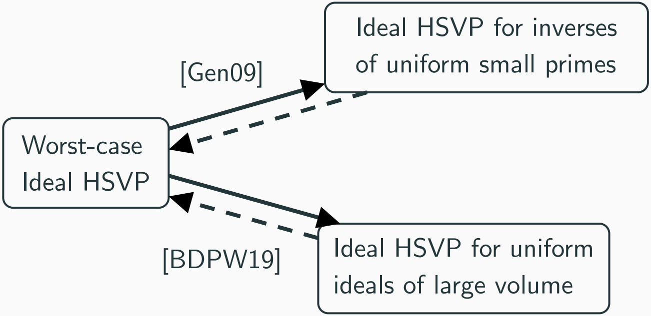
Average-case for D: Solve $\mathcal{P}$ for $I \leftarrow D$ *with non-negligible probability*.

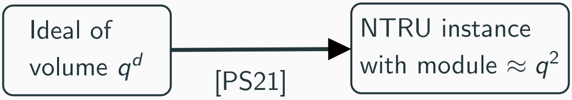
Average-case: "Find the secret key given a random public key".

Two reductions here:

- Worst-case to average case.
- Average case for D_1 to Average-case for D_2.

Prior Works on ideal-HSVP





And also: ideal-HSVP reduces to RLWE

Description of our work and motivation

Random version of $\mathrm{ideal\text{-}HSVP}$

$\mathcal{W}$-$\mathrm{ideal\text{-}HSVP}$: solving $\mathrm{ideal\text{-}HSVP}$ for a uniform element of $\mathcal{W}$.

Idea: $\mathcal{W}$ is the set of all public keys (a set of ideals).

Note: there are sets $\mathcal{W}$ such that $\mathcal{W}$-$\mathrm{ideal\text{-}HSVP}$ is easy [BGP22].

Description of our work and motivation

Random version of $\mathrm{ideal\text{-}HSVP}$

$\mathcal{W}$-$\mathrm{ideal\text{-}HSVP}$: solving $\mathrm{ideal\text{-}HSVP}$ for a uniform element of $\mathcal{W}$.

Idea: $\mathcal{W}$ is the set of all public keys (a set of ideals).

Note: there are sets $\mathcal{W}$ such that $\mathcal{W}$-$\mathrm{ideal\text{-}HSVP}$ is easy [BGP22].

We show that $\mathcal{P}^{-1}$-$\mathrm{ideal\text{-}HSVP}$ reduces to $\mathcal{P}$-$\mathrm{ideal\text{-}HSVP}$.

Two reasons

1. [Gen09]: $\mathrm{ideal\text{-}HSVP}$ (for all ideals) reduces to $\mathcal{P}^{-1}$-$\mathrm{ideal\text{-}HSVP}$, we complete this reduction.
2. The NTRU reduction from [PS21] works only for integral ideals.

In fact we prove a more general reduction:

$\mathrm{ideal\text{-}HSVP}(\mathcal{W}^{-1})$ reduces to $\mathrm{ideal\text{-}HSVP}(\mathcal{W}) + \mathrm{ideal\text{-}HSVP}(\mathcal{I}_{A,B})$

Prior work: Gentry’s reduction

Rounding ideals

Two types of ideals: *integral* $\mathfrak{a} \subseteq \mathcal{O}_K$ and *fractional*: $I \subseteq K$.
(And replete: $I = x \cdot \mathfrak{a} \subset K_\mathbb{R}$ with $x \in K_\mathbb{R}^\times$)

Note

If $\mathfrak{a}$ is integral, $\mathfrak{a}^{-1}$ is fractional.

How to round an ideal

Take $x \leftarrow I^{-1}$ with $x \sim \lambda \cdot (1, \ldots, 1)^T$ with λ large, then $x \cdot I \subseteq \mathcal{O}_K$ and:

$$s = \text{ideal-HSVP}(I) \overset{\sim}{\Longleftrightarrow} x \cdot s = \text{ideal-HSVP}(x \cdot I).$$

Rounding ideals

Two types of ideals: *integral* $\mathfrak{a} \subseteq \mathcal{O}_K$ and *fractional*: $I \subseteq K$.
(And replete: $I = x \cdot \mathfrak{a} \subset K_{\mathbb{R}}$ with $x \in K_{\mathbb{R}}^{\times}$)

Note

If $\mathfrak{a}$ is integral, $\mathfrak{a}^{-1}$ is fractional.

How to round an ideal

Take $x \leftarrow I^{-1}$ with $x \sim \lambda \cdot (1, \ldots, 1)^T$ with λ large, then $x \cdot I \subseteq \mathcal{O}_K$ and:

$$s = \text{ideal-HSVP}(I) \overset{\sim}{\Longleftrightarrow} x \cdot s = \text{ideal-HSVP}(x \cdot I).$$

Rounding allows to **randomize** our ideals (sample random x).

$$x \in I^{-1} \text{ is small} \Rightarrow x \cdot I \text{ has small volume.}$$

In [BDPW20], rounding is done with large elements (due to LLL).

Warming up: Gentry's reduction [Gen09]

The oracle $\mathcal{O}$ solves $\mathrm{ideal\text{-}HSVP}$ for $\mathfrak{p}^{-1}$ with $\mathfrak{p}$ uniform small prime.

Algorithm 2.1 Outline of the reduction of [Gen09]

Input: An ideal $I = \mathfrak{b}^{-1}$.
Output: $x \in I \setminus \{0\}$ small.

1: Let $x = 1$.
2: **while** $\mathfrak{b}$ is large **do**
3: **Round** $\mathfrak{b}$**:** sample v in $\mathfrak{b}^{-1}$, let $\mathfrak{a} = v \cdot \mathfrak{b} \subseteq \mathcal{O}_K$.
4: **Factor** $\mathfrak{a}$**:** write $\mathfrak{a} = \mathfrak{p}_1^{e_1} \cdot \ldots \cdot \mathfrak{p}_k^{e_k}$ with $\mathfrak{p}_i$ primes. (Quantum)
5: **Sample:** $\mathfrak{p}_i$ uniformly, and let $w = \mathcal{O}(\mathfrak{p}_i^{-1})$.
6: **Update:** $x \leftarrow w \cdot x$, $\mathfrak{b} \leftarrow (w) \cdot \mathfrak{b}$.
7: **Return** x

The rounding step ensure that $\mathfrak{p}_i$ is uniform in the set of prime ideals and then we can use $\mathcal{O}$. (In fact we have to use rejection sampling.)

Warming up: Gentry's reduction [Gen09]

The oracle $\mathcal{O}$ solves $\mathrm{ideal\text{-}HSVP}$ for $\mathfrak{p}^{-1}$ with $\mathfrak{p}$ uniform small prime.

Algorithm 2.1 Outline of the reduction of [Gen09]

Input: An ideal $I = \mathfrak{b}^{-1}$.
Output: $x \in I \setminus \{0\}$ small.

1: Let $x = 1$.
2: **while** $\mathfrak{b}$ is large **do**
3: **Round** $\mathfrak{b}$**:** sample v in $\mathfrak{b}^{-1}$, let $\mathfrak{a} = v \cdot \mathfrak{b} \subseteq \mathcal{O}_K$.
4: **Factor** $\mathfrak{a}$**:** write $\mathfrak{a} = \mathfrak{p}_1^{e_1} \cdot \ldots \cdot \mathfrak{p}_k^{e_k}$ with $\mathfrak{p}_i$ primes. (Quantum)
5: **Sample:** $\mathfrak{p}_i$ uniformly, and let $w = \mathcal{O}(\mathfrak{p}_i^{-1})$.
6: **Update:** $x \leftarrow w \cdot x$, $\mathfrak{b} \leftarrow (w) \cdot \mathfrak{b}$.
7: **Return** x

The rounding step ensure that $\mathfrak{p}_i$ is uniform in the set of prime ideals and then we can use $\mathcal{O}$. (In fact we have to use rejection sampling.)

Warming up: Gentry's reduction [Gen09]

The oracle $\mathcal{O}$ solves $\mathrm{ideal\text{-}HSVP}$ for $\mathfrak{p}^{-1}$ with $\mathfrak{p}$ uniform small prime.

Algorithm 2.1 Outline of the reduction of [Gen09]

Input: An ideal $I = \mathfrak{b}^{-1}$.

Output: $x \in I \setminus \{0\}$ small.

Let $x = 1$.
while $\mathfrak{b}$ is large **do**
    Round $\mathfrak{b}$**:** sample v in $\mathfrak{b}^{-1}$, let $\mathfrak{a} = v \cdot \mathfrak{b} \subseteq \mathcal{O}_K$.
    Factor $\mathfrak{a}$**:** write $\mathfrak{a} = \mathfrak{p}_1^{e_1} \cdot \ldots \cdot \mathfrak{p}_k^{e_k}$ with $\mathfrak{p}_i$ primes. (Quantum)
    Sample: $\mathfrak{p}_i$ uniformly, and let $w = \mathcal{O}(\mathfrak{p}_i^{-1})$.
    Update: $x \leftarrow w \cdot x$, $\mathfrak{b} \leftarrow (w) \cdot \mathfrak{b}$.
Return x

The rounding step ensure that $\mathfrak{p}_i$ is uniform in the set of prime ideals and then we can use $\mathcal{O}$. (In fact we have to use rejection sampling.)

Warming up: Gentry's reduction [Gen09]

The oracle $\mathcal{O}$ solves $\mathrm{ideal\text{-}HSVP}$ for $\mathfrak{p}^{-1}$ with $\mathfrak{p}$ uniform small prime.

Algorithm 2.1 Outline of the reduction of [Gen09]

Input: An ideal $I = \mathfrak{b}^{-1}$.

Output: $x \in I \setminus \{0\}$ small.

1: Let $x = 1$.
2: **while** $\mathfrak{b}$ is large **do**
3: **Round** $\mathfrak{b}$**:** sample v in $\mathfrak{b}^{-1}$, let $\mathfrak{a} = v \cdot \mathfrak{b} \subseteq \mathcal{O}_K$.
4: **Factor** $\mathfrak{a}$**:** write $\mathfrak{a} = \mathfrak{p}_1^{e_1} \cdot \ldots \cdot \mathfrak{p}_k^{e_k}$ with $\mathfrak{p}_i$ primes. (Quantum)
5: **Sample:** $\mathfrak{p}_i$ uniformly, and let $w = \mathcal{O}(\mathfrak{p}_i^{-1})$.
6: **Update:** $x \leftarrow w \cdot x$, $\mathfrak{b} \leftarrow (w) \cdot \mathfrak{b}$.
7: **Return** x

The rounding step ensure that $\mathfrak{p}_i$ is uniform in the set of prime ideals and then we can use $\mathcal{O}$. (In fact we have to use rejection sampling.)

Warming up: Gentry's reduction [Gen09]

The oracle $\mathcal{O}$ solves $\mathrm{ideal\text{-}HSVP}$ for $\mathfrak{p}^{-1}$ with $\mathfrak{p}$ uniform small prime.

Algorithm 2.1 Outline of the reduction of [Gen09]

Input: An ideal $I = \mathfrak{b}^{-1}$.
Output: $x \in I \setminus \{0\}$ small.

Let $x = 1$.
while $\mathfrak{b}$ is large **do**
    Round $\mathfrak{b}$**:** sample v in $\mathfrak{b}^{-1}$, let $\mathfrak{a} = v \cdot \mathfrak{b} \subseteq \mathcal{O}_K$.
    Factor $\mathfrak{a}$**:** write $\mathfrak{a} = \mathfrak{p}_1^{e_1} \cdot \ldots \cdot \mathfrak{p}_k^{e_k}$ with $\mathfrak{p}_i$ primes. (Quantum)
    Sample: $\mathfrak{p}_i$ uniformly, and let $w = \mathcal{O}(\mathfrak{p}_i^{-1})$.
    Update: $x \leftarrow w \cdot x$, $\mathfrak{b} \leftarrow (w) \cdot \mathfrak{b}$.
Return x

The rounding step ensure that $\mathfrak{p}_i$ is uniform in the set of prime ideals and then we can use $\mathcal{O}$. (In fact we have to use rejection sampling.)

Warming up: Gentry's reduction [Gen09]

The oracle $\mathcal{O}$ solves $\mathrm{ideal\text{-}HSVP}$ for $\mathfrak{p}^{-1}$ with $\mathfrak{p}$ uniform small prime.

Algorithm 2.1 Outline of the reduction of [Gen09]

Input: An ideal $I = \mathfrak{b}^{-1}$.
Output: $x \in I \setminus \{0\}$ small.

Let $x = 1$.
while $\mathfrak{b}$ is large **do**
    Round $\mathfrak{b}$**:** sample v in $\mathfrak{b}^{-1}$, let $\mathfrak{a} = v \cdot \mathfrak{b} \subseteq \mathcal{O}_K$.
    Factor $\mathfrak{a}$**:** write $\mathfrak{a} = \mathfrak{p}_1^{e_1} \cdot \ldots \cdot \mathfrak{p}_k^{e_k}$ with $\mathfrak{p}_i$ primes. (Quantum)
    Sample: $\mathfrak{p}_i$ uniformly, and let $w = \mathcal{O}(\mathfrak{p}_i^{-1})$.
    Update: $x \leftarrow w \cdot x$, $\mathfrak{b} \leftarrow (w) \cdot \mathfrak{b}$.
Return x

The rounding step ensure that $\mathfrak{p}_i$ is uniform in the set of prime ideals and then we can use $\mathcal{O}$. (In fact we have to use rejection sampling.)

Sampling ideals

How would you sample a big prime number in $[A, B]$?

You are not allowed to pre-compute the set of primes numbers in $[A, B]$!

Idea 1: rejection sampling

Sample N uniform in $[A, B]$ until it is prime, then output it.
$\rightarrow$ Not fitted our case, more on that later.

How would you sample a big prime number in $[A, B]$?

You are not allowed to pre-compute the set of primes numbers in $[A, B]$!

Idea 1: rejection sampling

Sample N uniform in $[A, B]$ until it is prime, then output it.
$\rightarrow$ Not fitted our case, more on that later.

Idea 2: factoring

Sample N uniform in $[A, B]$. Factor $N = \prod p_i^{e_i}$ and output a random $p_i \in [A, B]$.
$\rightarrow$ Need rejection sampling: 2 more frequent than 7919.
$\rightarrow$ It really looks like what we did in Gentry's reduction.

How would you sample a big prime number in $[A, B]$?

You are not allowed to pre-compute the set of primes numbers in $[A, B]$!

Idea 1: rejection sampling

Sample N uniform in $[A, B]$ until it is prime, then output it.
$\rightarrow$ Not fitted our case, more on that later.

Idea 2: factoring

Sample N uniform in $[A, B]$. Factor $N = \prod p_i^{e_i}$ and output a random $p_i \in [A, B]$.
$\rightarrow$ Need rejection sampling: 2 more frequent than 7919.
$\rightarrow$ It really looks like what we did in Gentry's reduction.

We are going to use Idea 2 in order to sample prime ideals with a trapdoor.

Sampling uniform prime ideals with trapdoor [PS21]

Algorithm 3.1 `SampleWithTrap` algorithm

Input: $2 \leq A < B$ integers

Output: $(\mathfrak{p}, x)$ such that $x \in \mathfrak{p}$ and $\mathcal{N}(\mathfrak{p}) \in [A, B]$.

1: **repeat**
2: Sample a small Gaussian x in $\mathcal{O}_K$. (Need a good basis of $\mathcal{O}_K$)
3: Factor (x): write $(x) = \mathfrak{p}_1^{e_1} \cdot \ldots \cdot \mathfrak{p}_k^{e_k}$ with $\mathfrak{p}_i$ primes. (Quantum)
4: **until** $\{\mathfrak{p}_i,\ \mathcal{N}(\mathfrak{p}_i) \in [A, B]\} \neq \emptyset$.
5: Pick: $\mathfrak{p} \leftarrow \{\mathfrak{p}_i,\ \mathcal{N}(\mathfrak{p}_i) \in [A, B]\}$ uniformly. (Rejection sampling here)
6: **Return** $(\mathfrak{p}, x)$

Sampling uniform prime ideals with trapdoor [PS21]

Algorithm 3.1 `SampleWithTrap` algorithm

Input: $2 \leq A < B$ integers

Output: $(\mathfrak{p}, x)$ such that $x \in \mathfrak{p}$ and $\mathcal{N}(\mathfrak{p}) \in [A, B]$.

repeat

  Sample a small Gaussian x in $\mathcal{O}_K$. (Need a good basis of $\mathcal{O}_K$)

  Factor (x): write $(x) = \mathfrak{p}_1^{e_1} \cdot \ldots \cdot \mathfrak{p}_k^{e_k}$ with $\mathfrak{p}_i$ primes. (Quantum)

until $\{\mathfrak{p}_i,\ \mathcal{N}(\mathfrak{p}_i) \in [A, B]\} \neq \emptyset$.

Pick: $\mathfrak{p} \leftarrow \{\mathfrak{p}_i,\ \mathcal{N}(\mathfrak{p}_i) \in [A, B]\}$ uniformly. (Rejection sampling here)

Return $(\mathfrak{p}, x)$

Sampling uniform prime ideals with trapdoor [PS21]

Algorithm 3.1 `SampleWithTrap` algorithm

Input: $2 \leq A < B$ integers

Output: $(\mathfrak{p}, x)$ such that $x \in \mathfrak{p}$ and $\mathcal{N}(\mathfrak{p}) \in [A, B]$.

1: **repeat**

2: Sample a small Gaussian x in $\mathcal{O}_K$. (Need a good basis of $\mathcal{O}_K$)

3: Factor (x): write $(x) = \mathfrak{p}_1^{e_1} \cdot \ldots \cdot \mathfrak{p}_k^{e_k}$ with $\mathfrak{p}_i$ primes. (Quantum)

4: **until** $\{\mathfrak{p}_i,\ \mathcal{N}(\mathfrak{p}_i) \in [A, B]\} \neq \emptyset$.

5: Pick: $\mathfrak{p} \leftarrow \{\mathfrak{p}_i,\ \mathcal{N}(\mathfrak{p}_i) \in [A, B]\}$ uniformly. (Rejection sampling here)

6: **Return** $(\mathfrak{p}, x)$

Sampling uniform prime ideals with trapdoor [PS21]

Algorithm 3.1 `SampleWithTrap` algorithm

Input: $2 \leq A < B$ integers

Output: $(\mathfrak{p}, x)$ such that $x \in \mathfrak{p}$ and $\mathcal{N}(\mathfrak{p}) \in [A, B]$.

1: **repeat**
2: Sample a small Gaussian x in $\mathcal{O}_K$. (Need a good basis of $\mathcal{O}_K$)
3: Factor (x): write $(x) = \mathfrak{p}_1^{e_1} \cdot \ldots \cdot \mathfrak{p}_k^{e_k}$ with $\mathfrak{p}_i$ primes. (Quantum)
4: **until** $\{\mathfrak{p}_i,\ \mathcal{N}(\mathfrak{p}_i) \in [A, B]\} \neq \emptyset$.
5: Pick: $\mathfrak{p} \leftarrow \{\mathfrak{p}_i,\ \mathcal{N}(\mathfrak{p}_i) \in [A, B]\}$ uniformly. (Rejection sampling here)
6: **Return** $(\mathfrak{p}, x)$

Sampling uniform prime ideals with trapdoor [PS21]

Algorithm 3.1 `SampleWithTrap` algorithm

Input: $2 \le A < B$ integers

Output: $(\mathfrak{p}, x)$ such that $x \in \mathfrak{p}$ and $\mathcal{N}(\mathfrak{p}) \in [A, B]$.

1: **repeat**
2: Sample a small Gaussian x in $\mathcal{O}_K$. (Need a good basis of $\mathcal{O}_K$)
3: Factor (x): write $(x) = \mathfrak{p}_1^{e_1} \cdot \ldots \cdot \mathfrak{p}_k^{e_k}$ with $\mathfrak{p}_i$ primes. (Quantum)
4: **until** $\{\mathfrak{p}_i,\ \mathcal{N}(\mathfrak{p}_i) \in [A, B]\} \neq \emptyset$.
5: Pick: $\mathfrak{p} \leftarrow \{\mathfrak{p}_i,\ \mathcal{N}(\mathfrak{p}_i) \in [A, B]\}$ uniformly. (Rejection sampling here)
6: **Return** $(\mathfrak{p}, x)$

Sampling uniform prime ideals with trapdoor [PS21]

Algorithm 3.1 `SampleWithTrap` algorithm

Input: $2 \leq A < B$ integers

Output: $(\mathfrak{p}, x)$ such that $x \in \mathfrak{p}$ and $\mathcal{N}(\mathfrak{p}) \in [A, B]$.

1: **repeat**

2: Sample a small Gaussian x in $\mathcal{O}_K$. (Need a good basis of $\mathcal{O}_K$)

3: Factor (x): write $(x) = \mathfrak{p}_1^{e_1} \cdot \ldots \cdot \mathfrak{p}_k^{e_k}$ with $\mathfrak{p}_i$ primes. (Quantum)

4: **until** $\{\mathfrak{p}_i,\ \mathcal{N}(\mathfrak{p}_i) \in [A, B]\} \neq \emptyset$.

5: Pick: $\mathfrak{p} \leftarrow \{\mathfrak{p}_i,\ \mathcal{N}(\mathfrak{p}_i) \in [A, B]\}$ uniformly. (Rejection sampling here)

6: **Return** $(\mathfrak{p}, x)$

Theorem

This algorithms runs in quantum poly-time and outputs $\mathfrak{p}$ almost uniform in $\mathcal{P}_{A,B}$ along with small $x \in \mathfrak{p} \setminus \{0\}$.

Arakelov ideal sampling [BDPW20, Boe22]

Allows to sample uniform ideals:

Algorithm 3.2 `ArakelovSampling` algorithm

Output: An ideal $\mathfrak{b}$

1: Let $\mathfrak{q}$ an uniform small prime ideal.
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}/\mathcal{N}(\mathfrak{q})^{1/d}$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

`ArakelovSampling` output uniform integral ideals of norm $\approx r^d$ for $r = 2^{O(d)}$.

Arakelov ideal sampling [BDPW20, Boe22]

Allows to sample uniform ideals:

Algorithm 3.2 `ArakelovSampling` algorithm

Output: An ideal $\mathfrak{b}$

1: Let $\mathfrak{q}$ an uniform small prime ideal.
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}/\mathcal{N}(\mathfrak{q})^{1/d}$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

`ArakelovSampling` output uniform integral ideals of norm $\approx r^d$ for $r = 2^{O(d)}$.

Arakelov ideal sampling [BDPW20, Boe22]

Allows to sample uniform ideals:

Algorithm 3.2 `ArakelovSampling` algorithm

Output: An ideal $\mathfrak{b}$

1: Let $\mathfrak{q}$ an uniform small prime ideal.
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}/\mathcal{N}(\mathfrak{q})^{1/d}$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

`ArakelovSampling` output uniform integral ideals of norm $\approx r^d$ for $r = 2^{O(d)}$.

Arakelov ideal sampling [BDPW20, Boe22]

Allows to sample uniform ideals:

Algorithm 3.2 `ArakelovSampling` algorithm

Output: An ideal $\mathfrak{b}$

1: Let $\mathfrak{q}$ an uniform small prime ideal.
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}/\mathcal{N}(\mathfrak{q})^{1/d}$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

`ArakelovSampling` output uniform integral ideals of norm $\approx r^d$ for $r = 2^{O(d)}$.

Arakelov ideal sampling [BDPW20, Boe22]

Allows to sample uniform ideals:

Algorithm 3.2 `ArakelovSampling` algorithm

Output: An ideal $\mathfrak{b}$

1: Let $\mathfrak{q}$ an uniform small prime ideal.
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}/\mathcal{N}(\mathfrak{q})^{1/d}$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

`ArakelovSampling` output uniform integral ideals of norm $\approx r^d$ for $r = 2^{O(d)}$.

Arakelov ideal sampling [BDPW20, Boe22]

Allows to sample uniform ideals:

Algorithm 3.2 `ArakelovSampling` algorithm

Output: An ideal $\mathfrak{b}$

1: Let $\mathfrak{q}$ an uniform small prime ideal.
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}/\mathcal{N}(\mathfrak{q})^{1/d}$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

`ArakelovSampling` output uniform integral ideals of norm $\approx r^d$ for $r = 2^{O(d)}$.

What if we want to sample with a trapdoor inside?

We modify our algorithm to output some small element in $\mathfrak{b}^{-1}$

Algorithm 3.3 `ArakelovSampling`′ algorithm

Output: An ideal $\mathfrak{b}$ and $y \in \mathfrak{b}^{-1}$.

1: Let $\mathfrak{q}$ an uniform small prime ideal.
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}/\mathcal{N}(\mathfrak{q})^{1/d}$
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$.
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

What if we want to sample with a trapdoor inside?

We modify our algorithm to output some small element in $\mathfrak{b}^{-1}$

Algorithm 3.3 `ArakelovSampling'` algorithm

Output: An ideal $\mathfrak{b}$ and $y \in \mathfrak{b}^{-1}$.

1: Let $\mathfrak{q}$ an uniform small prime ideal.
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q} / \mathcal{N}(\mathfrak{q})^{1/d}$
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$.
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

What if we want to sample with a trapdoor inside?

We modify our algorithm to output some small element in $\mathfrak{b}^{-1}$

Algorithm 3.3 `ArakelovSampling'` algorithm

Output: An ideal $\mathfrak{b}$ and $y \in \mathfrak{b}^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow$ `SampleWithTrap`$(\cdot)$. (Quantum)
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}/\mathcal{N}(\mathfrak{q})^{1/d}$
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_{\infty}(r) \bigcap I\right)$.
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

What if we want to sample with a trapdoor inside?

We modify our algorithm to output some small element in $\mathfrak{b}^{-1}$

Algorithm 3.3 `ArakelovSampling`′ algorithm

Output: An ideal $\mathfrak{b}$ and $y \in \mathfrak{b}^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)

2: Sample a small continuous Gaussian ζ and a uniform rotation u.

3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q} / \mathcal{N}(\mathfrak{q})^{1/d}$

4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$.

5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

What if we want to sample with a trapdoor inside?

We modify our algorithm to output some small element in $\mathfrak{b}^{-1}$

Algorithm 3.3 `ArakelovSampling`′ algorithm

Output: An ideal $\mathfrak{b}$ and $y \in \mathfrak{b}^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}/\mathcal{N}(\mathfrak{q})^{1/d}$
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$.
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

What if we want to sample with a trapdoor inside?

We modify our algorithm to output some small element in $\mathfrak{b}^{-1}$

Algorithm 3.3 `ArakelovSampling′` algorithm

Output: An ideal $\mathfrak{b}$ and $y \in \mathfrak{b}^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)

2: Sample a small continuous Gaussian ζ and a uniform rotation u.

3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}/\mathcal{N}(\mathfrak{q})^{1/d}$ and $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}}/\mathcal{N}(\mathfrak{q})^{1/d} \in I$.

4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$.

5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

What if we want to sample with a trapdoor inside?

We modify our algorithm to output some small element in $\mathfrak{b}^{-1}$

Algorithm 3.3 `ArakelovSampling`′ algorithm

Output: An ideal $\mathfrak{b}$ and $y \in \mathfrak{b}^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q} / \mathcal{N}(\mathfrak{q})^{1/d}$ and $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}} / \mathcal{N}(\mathfrak{q})^{1/d} \in I$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$.
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

What if we want to sample with a trapdoor inside?

We modify our algorithm to output some small element in $\mathfrak{b}^{-1}$

Algorithm 3.3 `ArakelovSampling′` algorithm

Output: An ideal $\mathfrak{b}$ and $y \in \mathfrak{b}^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}/\mathcal{N}(\mathfrak{q})^{1/d}$ and $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}}/\mathcal{N}(\mathfrak{q})^{1/d} \in I$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$.
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$

What if we want to sample with a trapdoor inside?

We modify our algorithm to output some small element in $\mathfrak{b}^{-1}$

Algorithm 3.3 `ArakelovSampling'` algorithm

Output: An ideal $\mathfrak{b}$ and $y \in \mathfrak{b}^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}/\mathcal{N}(\mathfrak{q})^{1/d}$ and $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}}/\mathcal{N}(\mathfrak{q})^{1/d} \in I$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$.
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$ and $y = x^{-1} \cdot s_I$.

What if we want to sample with a trapdoor inside?

We modify our algorithm to output some small element in $\mathfrak{b}^{-1}$

Algorithm 3.3 `ArakelovSampling'` algorithm

Output: An ideal $\mathfrak{b}$ and $y \in \mathfrak{b}^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q} / \mathcal{N}(\mathfrak{q})^{1/d}$ and $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}} / \mathcal{N}(\mathfrak{q})^{1/d} \in I$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$.
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$ and $y = x^{-1} \cdot s_I$.

Drawback

The element $y = x^{-1} \cdot s_I$ can be very large compared to $\mathcal{N}(\mathfrak{b}^{-1})^{1/d}$.

What if we want to sample with a trapdoor inside?

We modify our algorithm to output some small element in $\mathfrak{b}^{-1}$

Algorithm 3.3 `ArakelovSampling`′ algorithm

Output: An ideal $\mathfrak{b}$ and $y \in \mathfrak{b}^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q}/\mathcal{N}(\mathfrak{q})^{1/d}$ and $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}}/\mathcal{N}(\mathfrak{q})^{1/d} \in I$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$.
5: **Return** $\mathfrak{b} = x \cdot I^{-1}$ and $y = x^{-1} \cdot s_I$.

Drawback

The element $y = x^{-1} \cdot s_I$ can be very large compared to $\mathcal{N}(\mathfrak{b}^{-1})^{1/d}$.

$\rightarrow$ This happens if x is **unbalanced**

Some details on `ArakelovSampling`

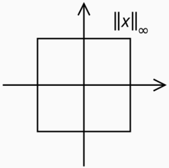


Figure 2: $\mathcal{B}_\infty(r)$

1. We pick $I \approx \mathfrak{q}/\mathcal{N}(\mathfrak{q})^{1/d}$.
2. We sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$.
3. We return $\mathfrak{b} = x \cdot I^{-1}$.

Necessary for uniform $\mathfrak{b}$

1. $|\mathcal{B}_\infty(r) \bigcap I|$ do not depend on I (too much).
2. $\mathrm{Vol}(\mathrm{Log}(\mathcal{B}_\infty(r)) \bigcap \{\sum x_i = t\})$ is $\approx$ constant for $t \in [A, B]$.

Some details on `ArakelovSampling`

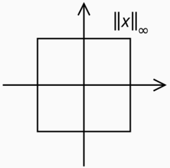


Figure 2: $\mathcal{B}_\infty(r)$

1. We pick $I \approx \mathfrak{q}/\mathcal{N}(\mathfrak{q})^{1/d}$.
2. We sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_\infty(r) \bigcap I\right)$.
3. We return $\mathfrak{b} = x \cdot I^{-1}$.

Necessary for uniform $\mathfrak{b}$

1. $|\mathcal{B}_\infty(r) \bigcap I|$ do not depend on I (too much).
2. $\mathrm{Vol}(\mathrm{Log}(\mathcal{B}_\infty(r)) \bigcap \{\sum x_i = t\})$ is $\approx$ constant for $t \in [A, B]$.

Drawback

There are (a non-negligible proportion of) $x \in \mathcal{B}_\infty(r)$ with $\left\|x^{-1}\right\|$ very large.

Main contribution:
$\mathcal{P}^{-1}$-ideal-SVP **to** $\mathcal{P}$-ideal-SVP

First contribution: Generalized Arakelov ideal sampling

We generalize the approach of [BDPW20, Boe22]:

Algorithm 4.1 `SampleIdeal`$_{\mathcal{B}_{A,B}}$ algorithm

Input: $\mathfrak{a}$ an ideal, $s_{\mathfrak{a}} \in \mathfrak{a}$ small, $\mathcal{B}_{A,B} \subset K_{\mathbb{R}}$ a **well chosen** set.

Output: $(\mathfrak{b}, y)$ such that $y \in (\mathfrak{b} \cdot \mathfrak{a})^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)

2: Sample a small continuous Gaussian ζ and a uniform rotation u.

3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q} \cdot \mathfrak{a}$ and $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}} \cdot s_{\mathfrak{a}} \in I$.

4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_{A,B} \bigcap I\right)$ using s_I .

5: **Return** $(\mathfrak{b} = x \cdot I^{-1}, y = x^{-1} \cdot s_I \cdot v_{\mathfrak{q}})$

(Normalization factors omitted)

First contribution: Generalized Arakelov ideal sampling

We generalize the approach of [BDPW20, Boe22]:

Algorithm 4.1 `SampleIdeal`$_{\mathcal{B}_{A,B}}$ algorithm

Input: $\mathfrak{a}$ an ideal, $s_{\mathfrak{a}} \in \mathfrak{a}$ small, $\mathcal{B}_{A,B} \subset K_{\mathbb{R}}$ a **well chosen** set.

Output: $(\mathfrak{b}, y)$ such that $y \in (\mathfrak{b} \cdot \mathfrak{a})^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)

2: Sample a small continuous Gaussian ζ and a uniform rotation u.

3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q} \cdot \mathfrak{a}$ and $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}} \cdot s_{\mathfrak{a}} \in I$.

4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_{A,B} \bigcap I\right)$ using s_I .

5: **Return** $\left(\mathfrak{b} = x \cdot I^{-1}, y = x^{-1} \cdot s_I \cdot v_{\mathfrak{q}}\right)$

(Normalization factors omitted)

First contribution: Generalized Arakelov ideal sampling

We generalize the approach of [BDPW20, Boe22]:

Algorithm 4.1 $\texttt{SampleIdeal}_{\mathcal{B}_{A,B}}$ algorithm

Input: $\mathfrak{a}$ an ideal, $s_\mathfrak{a} \in \mathfrak{a}$ small, $\mathcal{B}_{A,B} \subset K_\mathbb{R}$ a **well chosen** set.

Output: $(\mathfrak{b}, y)$ such that $y \in (\mathfrak{b} \cdot \mathfrak{a})^{-1}$.

1: Let $(\mathfrak{q}, v_\mathfrak{q}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)

2: Sample a small continuous Gaussian ζ and a uniform rotation u.

3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q} \cdot \mathfrak{a}$ and $s_I = \exp(\zeta) \cdot u \cdot s_\mathfrak{q} \cdot s_\mathfrak{a} \in I$.

4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_{A,B} \bigcap I\right)$ using s_I .

5: **Return** $\left(\mathfrak{b} = x \cdot I^{-1}, y = x^{-1} \cdot s_I \cdot v_\mathfrak{q}\right)$

(Normalization factors omitted)

First contribution: Generalized Arakelov ideal sampling

We generalize the approach of [BDPW20, Boe22]:

Algorithm 4.1 `SampleIdeal`$_{\mathcal{B}_{A,B}}$ algorithm

Input: $\mathfrak{a}$ an ideal, $s_{\mathfrak{a}} \in \mathfrak{a}$ small, $\mathcal{B}_{A,B} \subset K_{\mathbb{R}}$ a **well chosen** set.
Output: $(\mathfrak{b}, y)$ such that $y \in (\mathfrak{b} \cdot \mathfrak{a})^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q} \cdot \mathfrak{a}$ and $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}} \cdot s_{\mathfrak{a}} \in I$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_{A,B} \bigcap I\right)$ using s_I .
5: **Return** $(\mathfrak{b} = x \cdot I^{-1}, y = x^{-1} \cdot s_I \cdot v_{\mathfrak{q}})$

(Normalization factors omitted)

First contribution: Generalized Arakelov ideal sampling

We generalize the approach of [BDPW20, Boe22]:

Algorithm 4.1 $\texttt{SampleIdeal}_{\mathcal{B}_{A,B}}$ algorithm

Input: $\mathfrak{a}$ an ideal, $s_{\mathfrak{a}} \in \mathfrak{a}$ small, $\mathcal{B}_{A,B} \subset K_{\mathbb{R}}$ a **well chosen** set.
Output: $(\mathfrak{b}, y)$ such that $y \in (\mathfrak{b} \cdot \mathfrak{a})^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)
2: Sample a small continuous Gaussian ζ and a uniform rotation u.
3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q} \cdot \mathfrak{a}$ and $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}} \cdot s_{\mathfrak{a}} \in I$.
4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_{A,B} \bigcap I\right)$ using s_I .
5: **Return** $(\mathfrak{b} = x \cdot I^{-1}, y = x^{-1} \cdot s_I \cdot v_{\mathfrak{q}})$

(Normalization factors omitted)

First contribution: Generalized Arakelov ideal sampling

We generalize the approach of [BDPW20, Boe22]:

Algorithm 4.1 `SampleIdeal`$_{\mathcal{B}_{A,B}}$ algorithm

Input: $\mathfrak{a}$ an ideal, $s_{\mathfrak{a}} \in \mathfrak{a}$ small, $\mathcal{B}_{A,B} \subset K_{\mathbb{R}}$ a **well chosen** set.

Output: $(\mathfrak{b}, y)$ such that $y \in (\mathfrak{b} \cdot \mathfrak{a})^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)

2: Sample a small continuous Gaussian ζ and a uniform rotation u.

3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q} \cdot \mathfrak{a}$ and $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}} \cdot s_{\mathfrak{a}} \in I$.

4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_{A,B} \bigcap I\right)$ using s_I .

5: **Return** $\left(\mathfrak{b} = x \cdot I^{-1}, y = x^{-1} \cdot s_I \cdot v_{\mathfrak{q}}\right)$

(Normalization factors omitted)

First contribution: Generalized Arakelov ideal sampling

We generalize the approach of [BDPW20, Boe22]:

Algorithm 4.1 $\texttt{SampleIdeal}_{\mathcal{B}_{A,B}}$ algorithm

Input: $\mathfrak{a}$ an ideal, $s_{\mathfrak{a}} \in \mathfrak{a}$ small, $\mathcal{B}_{A,B} \subset K_{\mathbb{R}}$ a **well chosen** set.

Output: $(\mathfrak{b}, y)$ such that $y \in (\mathfrak{b} \cdot \mathfrak{a})^{-1}$.

1: Let $(\mathfrak{q}, v_{\mathfrak{q}}) \leftarrow \texttt{SampleWithTrap}(\cdot)$. (Quantum)

2: Sample a small continuous Gaussian ζ and a uniform rotation u.

3: Let $I = \exp(\zeta) \cdot u \cdot \mathfrak{q} \cdot \mathfrak{a}$ and $s_I = \exp(\zeta) \cdot u \cdot s_{\mathfrak{q}} \cdot s_{\mathfrak{a}} \in I$.

4: Sample $x \hookleftarrow \mathcal{U}\left(\mathcal{B}_{A,B} \bigcap I\right)$ using s_I .

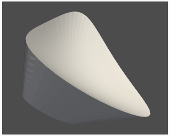
5: **Return** $(\mathfrak{b} = x \cdot I^{-1}, y = x^{-1} \cdot s_I \cdot v_{\mathfrak{q}})$

(Normalization factors omitted)

Theorem

Let $(\mathfrak{b}, y) = \texttt{SampleIdeal}_{\mathcal{B}_{A,B}}(\mathfrak{a}, s_{\mathfrak{a}}, A, B)$.
If $\mathcal{B}_{A,B}$ *is* **well chosen** *then* $\mathfrak{b}$ *is almost uniform in* $\mathcal{I}_{A,B}$ *and* y *is small.*

What does "well chosen" means?



- $|\mathcal{B}_{A,B} \bigcap \mathfrak{a}|$ do not depend on $\mathfrak{a}$ (too much).
- $\mathrm{Vol}(\mathrm{Log}(\mathcal{B}_{A,B}) \bigcap \{\sum x_i = t\})$ is constant for $t \in [A, B]$.
- Its elements must be balanced.

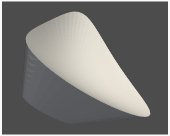
Balanced elements (for Minkowski embedding)

$x \in K$ is balanced if for all i,

$$1/\eta \leq x_i / \prod_j x_j^{1/d} \leq \eta.$$

This is the same as saying $x \approx \mathcal{N}(x)^{1/d} \cdot (1, \ldots, 1)$.

What does "well chosen" means?



- $|\mathcal{B}_{A,B} \bigcap \mathfrak{a}|$ do not depend on $\mathfrak{a}$ (too much).
- $\mathrm{Vol}(\mathrm{Log}(\mathcal{B}_{A,B}) \bigcap \{\sum x_i = t\})$ is constant for $t \in [A, B]$.
- Its elements must be balanced.

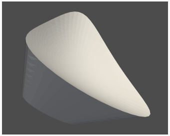
Balanced elements (for Minkowski embedding)

$x \in K$ is balanced if for all i,

$$1/\eta \leq x_i / \prod_j x_j^{1/d} \leq \eta.$$

This is the same as saying $x \approx \mathcal{N}(x)^{1/d} \cdot (1, \ldots, 1)$.

What does "well chosen" means?



- $|\mathcal{B}_{A,B} \bigcap \mathfrak{a}|$ do not depend on $\mathfrak{a}$ (too much).
- $\mathrm{Vol}(\mathrm{Log}(\mathcal{B}_{A,B}) \bigcap \{\sum x_i = t\})$ is constant for $t \in [A, B]$.
- Its elements must be balanced.

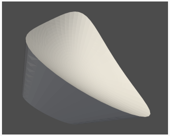
Balanced elements **(for Minkowski embedding)**

$x \in K$ is balanced if for all i,

$$1/\eta \leq x_i / \prod_j x_j^{1/d} \leq \eta.$$

This is the same as saying $x \approx \mathcal{N}(x)^{1/d} \cdot (1, \ldots, 1)$.

What does "well chosen" means?



- $|\mathcal{B}_{A,B} \bigcap \mathfrak{a}|$ do not depend on $\mathfrak{a}$ (too much).
- $\mathrm{Vol}(\mathrm{Log}(\mathcal{B}_{A,B}) \bigcap \{\sum x_i = t\})$ is constant for $t \in [A, B]$.
- Its elements must be balanced.

Balanced elements (for Minkowski embedding)

$x \in K$ is balanced if for all i,

$$1/\eta \leq x_i / \prod_j x_j^{1/d} \leq \eta.$$

This is the same as saying $x \approx \mathcal{N}(x)^{1/d} \cdot (1, \ldots, 1)$.

In [BDPW20]: $\mathcal{B}_\infty(r)$: verify points 1 and 2 but not 3!

Our shape

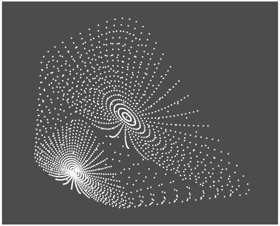
Reminder: conditions for being **well chosen**:

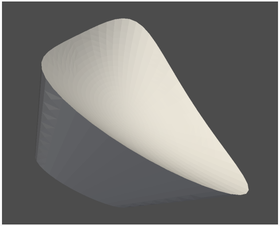
- $|\mathcal{B}_{A,B} \bigcap \mathfrak{a}|$ do not depend on $\mathfrak{a}$ (too much).
- $\mathrm{Vol}(\mathrm{Log}(\mathcal{B}_{A,B}) \bigcap \{\sum x_i = t\})$ is constant for $t \in [A, B]$.
- Its elements must be balanced.

Our shape

Reminder: conditions for being **well chosen**:

- $|\mathcal{B}_{A,B} \bigcap \mathfrak{a}|$ do not depend on $\mathfrak{a}$ (too much).
- $\mathrm{Vol}(\mathrm{Log}(\mathcal{B}_{A,B}) \bigcap \{\sum x_i = t\})$ is constant for $t \in [A, B]$.
- Its elements must be balanced.



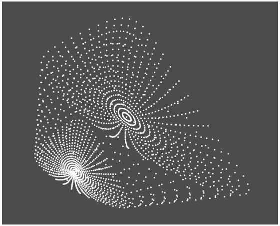


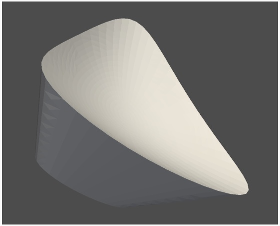
$$\mathcal{B}^{\eta}_{A,B} = \left\{ x \in K_{\mathbb{R}}, \quad |\mathcal{N}(x)| \in [A, B], \quad \left\| \mathrm{Log}\left(\frac{x}{\mathcal{N}(x)^{1/d}} \right) \right\|_2 \leq \log(\eta) \right\}$$

Our shape

Reminder: conditions for being **well chosen**:

- $|\mathcal{B}_{A,B} \bigcap \mathfrak{a}|$ do not depend on $\mathfrak{a}$ (too much).
- $\mathrm{Vol}(\mathrm{Log}(\mathcal{B}_{A,B}) \bigcap \{\sum x_i = t\})$ is constant for $t \in [A, B]$.
- Its elements must be balanced.



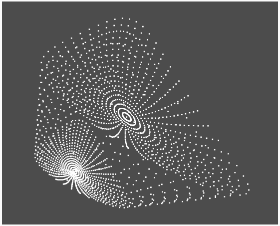


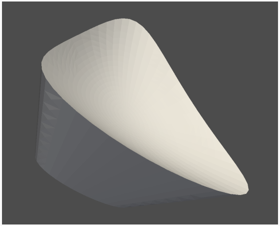
$$\mathcal{B}_{A,B}^{\eta} = \left\{ x \in K_{\mathbb{R}}, \quad |\mathcal{N}(x)| \in [A, B], \quad \left\| \mathrm{Log}\left(\frac{x}{\mathcal{N}(x)^{1/d}}\right) \right\|_2 \leq \log(\eta) \right\}$$

Our shape

Reminder: conditions for being **well chosen**:

- $|\mathcal{B}_{A,B} \bigcap \mathfrak{a}|$ do not depend on $\mathfrak{a}$ (too much).
- $\mathrm{Vol}(\mathrm{Log}(\mathcal{B}_{A,B}) \bigcap \{\sum x_i = t\})$ is constant for $t \in [A, B]$.
- Its elements must be balanced.



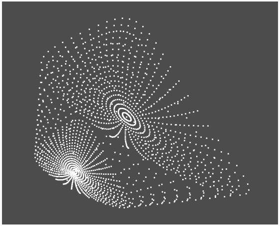


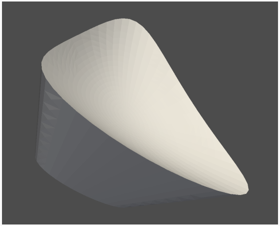
$$\mathcal{B}^{\eta}_{A,B} = \left\{ x \in K_{\mathbb{R}}, \quad |\mathcal{N}(x)| \in [A, B], \quad \left\| \mathrm{Log}\left(\frac{x}{\mathcal{N}(x)^{1/d}}\right) \right\|_2 \leq \log(\eta) \right\}$$

Our shape

Reminder: conditions for being **well chosen**:

- $|\mathcal{B}_{A,B} \bigcap \mathfrak{a}|$ do not depend on $\mathfrak{a}$ (too much).
- $\mathrm{Vol}(\mathrm{Log}(\mathcal{B}_{A,B}) \bigcap \{\sum x_i = t\})$ is constant for $t \in [A, B]$.
- Its elements must be balanced.





$$\mathcal{B}^{\eta}_{A,B} = \left\{ x \in K_{\mathbb{R}}, \quad |\mathcal{N}(x)| \in [A, B], \quad \left\| \mathrm{Log}\left(\frac{x}{\mathcal{N}(x)^{1/d}} \right) \right\|_2 \leq \log(\eta) \right\}$$

Reminder: $\texttt{SampleIdeal}_{\mathcal{B}_{A,B}}$

The algorithm $\texttt{SampleIdeal}_{\mathcal{B}_{A,B}}$:

1. Takes as input $\mathfrak{a} \subseteq \mathcal{O}_K$ and $s_{\mathfrak{a}} \in \mathfrak{a}$ small.
2. Output $\mathfrak{b} \subseteq \mathcal{O}_K$ uniform and $y \in \mathfrak{b}^{-1} \cdot \mathfrak{a}^{-1}$ small.

Reminder: $\texttt{SampleIdeal}_{\mathcal{B}_{A,B}}$

The algorithm $\texttt{SampleIdeal}_{\mathcal{B}_{A,B}}$:

1. Takes as input $\mathfrak{a} \subseteq \mathcal{O}_K$ and $s_\mathfrak{a} \in \mathfrak{a}$ small.
2. Output $\mathfrak{b} \subseteq \mathcal{O}_K$ uniform and $y \in \mathfrak{b}^{-1} \cdot \mathfrak{a}^{-1}$ small.

Now if we get in $s_\mathfrak{b} \in \mathfrak{b}$ small, we have that $s_\mathfrak{b} \cdot y$ is small and

$$s_\mathfrak{b} \cdot y \in \mathfrak{b} \cdot \mathfrak{b}^{-1} \cdot \mathfrak{a}^{-1} = \mathfrak{a}^{-1}$$

Reminder: $\texttt{SampleIdeal}_{\mathcal{B}_{A,B}}$

The algorithm $\texttt{SampleIdeal}_{\mathcal{B}_{A,B}}$:

1. Takes as input $\mathfrak{a} \subseteq \mathcal{O}_K$ and $s_\mathfrak{a} \in \mathfrak{a}$ small.
2. Output $\mathfrak{b} \subseteq \mathcal{O}_K$ uniform and $y \in \mathfrak{b}^{-1} \cdot \mathfrak{a}^{-1}$ small.

Now if we get in $s_\mathfrak{b} \in \mathfrak{b}$ small, we have that $s_\mathfrak{b} \cdot y$ is small and

$$s_\mathfrak{b} \cdot y \in \mathfrak{b} \cdot \mathfrak{b}^{-1} \cdot \mathfrak{a}^{-1} = \mathfrak{a}^{-1}$$

$$\boxed{\text{ideal-HSVP}(\mathfrak{a}) + \text{ideal-HSVP}(\mathfrak{b}) \xrightarrow{\texttt{SampleIdeal}_{\mathcal{B}_{A,B}}} \text{ideal-HSVP}(\mathfrak{a}^{-1})}$$

Our $\mathcal{P}^{-1}$-ideal-SVP to $\mathcal{P}$-ideal-SVP reduction

The oracle $\mathcal{O}$ solves $\mathrm{ideal\text{-}HSVP}$ for $\mathfrak{p}$ uniform prime of norm in $[A, B]$.

Algorithm 4.2 Outline of the $\mathcal{P}^{-1}$-ideal-SVP to $\mathcal{P}$-ideal-SVP reduction

Input: An ideal $I = \mathfrak{p}^{-1}$ with $\mathfrak{p}$ uniform prime of norm in $[A, B]$.

Output: $x \in I \setminus \{0\}$ small.

1: Let $s_{\mathfrak{p}} = \mathcal{O}(\mathfrak{p})$. ($\mathfrak{p}$ is uniform)

2: Let $(\mathfrak{b}, y) = \mathtt{SampleIdeal}_{A,B}(\mathfrak{p}, s_{\mathfrak{p}})$

3: **if** $\mathfrak{b}$ is not prime. (with probability $(\mathrm{poly} \cdot \rho_K)^{-1}$) **then**

4: Fail.

5: Let $s_{\mathfrak{b}} = \mathcal{O}(\mathfrak{b})$.

6: **Return** $\underbrace{s_{\mathfrak{b}}}_{\text{small}} \cdot \underbrace{y}_{\text{small}}$.

Our $\mathcal{P}^{-1}$-ideal-SVP to $\mathcal{P}$-ideal-SVP reduction

The oracle $\mathcal{O}$ solves $\mathrm{ideal\text{-}HSVP}$ for $\mathfrak{p}$ uniform prime of norm in $[A, B]$.

Algorithm 4.2 Outline of the $\mathcal{P}^{-1}$-ideal-SVP to $\mathcal{P}$-ideal-SVP reduction

Input: An ideal $I = \mathfrak{p}^{-1}$ with $\mathfrak{p}$ uniform prime of norm in $[A, B]$.

Output: $x \in I \setminus \{0\}$ small.

1: Let $s_{\mathfrak{p}} = \mathcal{O}(\mathfrak{p})$. ($\mathfrak{p}$ is uniform)

2: Let $(\mathfrak{b}, y) = \mathtt{SampleIdeal}_{A,B}(\mathfrak{p}, s_{\mathfrak{p}})$

3: **if** $\mathfrak{b}$ is not prime. (with probability $(\mathrm{poly} \cdot \rho_K)^{-1}$) **then**

4: Fail.

5: Let $s_{\mathfrak{b}} = \mathcal{O}(\mathfrak{b})$.

6: **Return** $\underbrace{s_{\mathfrak{b}}}_{\text{small}} \cdot \underbrace{y}_{\text{small}}$.

Our $\mathcal{P}^{-1}$-ideal-SVP to $\mathcal{P}$-ideal-SVP reduction

The oracle $\mathcal{O}$ solves ideal-HSVP for $\mathfrak{p}$ uniform prime of norm in $[A, B]$.

Algorithm 4.2 Outline of the $\mathcal{P}^{-1}$-ideal-SVP to $\mathcal{P}$-ideal-SVP reduction

Input: An ideal $I = \mathfrak{p}^{-1}$ with $\mathfrak{p}$ uniform prime of norm in $[A, B]$.

Output: $x \in I \setminus \{0\}$ small.

1: Let $s_{\mathfrak{p}} = \mathcal{O}(\mathfrak{p})$. ($\mathfrak{p}$ is uniform)

2: Let $(\mathfrak{b}, y) = \texttt{SampleIdeal}_{A,B}(\mathfrak{p}, s_{\mathfrak{p}})$

3: **if** $\mathfrak{b}$ is not prime. (with probability $(\text{poly} \cdot \rho_K)^{-1}$) **then**

4: Fail.

5: Let $s_{\mathfrak{b}} = \mathcal{O}(\mathfrak{b})$.

6: **Return** $\underbrace{s_{\mathfrak{b}}}_{\text{small}} \cdot \underbrace{y}_{\text{small}}$.

Our $\mathcal{P}^{-1}$-ideal-SVP to $\mathcal{P}$-ideal-SVP reduction

The oracle $\mathcal{O}$ solves ideal-HSVP for $\mathfrak{p}$ uniform prime of norm in $[A, B]$.

Algorithm 4.2 Outline of the $\mathcal{P}^{-1}$-ideal-SVP to $\mathcal{P}$-ideal-SVP reduction

Input: An ideal $I = \mathfrak{p}^{-1}$ with $\mathfrak{p}$ uniform prime of norm in $[A, B]$.

Output: $x \in I \setminus \{0\}$ small.

Let $s_{\mathfrak{p}} = \mathcal{O}(\mathfrak{p})$. ($\mathfrak{p}$ is uniform)

Let $(\mathfrak{b}, y) = \texttt{SampleIdeal}_{A,B}(\mathfrak{p}, s_{\mathfrak{p}})$

if $\mathfrak{b}$ is not prime. (with probability $(\text{poly} \cdot \rho_K)^{-1}$) **then**

    Fail.

Let $s_{\mathfrak{b}} = \mathcal{O}(\mathfrak{b})$.

Return $\underbrace{s_{\mathfrak{b}}}_{\text{small}} \cdot \underbrace{y}_{\text{small}}$.

Our $\mathcal{P}^{-1}$-ideal-SVP to $\mathcal{P}$-ideal-SVP reduction

The oracle $\mathcal{O}$ solves $\mathrm{ideal\text{-}HSVP}$ for $\mathfrak{p}$ uniform prime of norm in $[A, B]$.

Algorithm 4.2 Outline of the $\mathcal{P}^{-1}$-ideal-SVP to $\mathcal{P}$-ideal-SVP reduction

---

Input: An ideal $I = \mathfrak{p}^{-1}$ with $\mathfrak{p}$ uniform prime of norm in $[A, B]$.

Output: $x \in I \setminus \{0\}$ small.

1: Let $s_{\mathfrak{p}} = \mathcal{O}(\mathfrak{p})$. ($\mathfrak{p}$ is uniform)

2: Let $(\mathfrak{b}, y) = \texttt{SampleIdeal}_{A,B}(\mathfrak{p}, s_{\mathfrak{p}})$

3: **if** $\mathfrak{b}$ is not prime. (with probability $(\mathrm{poly} \cdot \rho_K)^{-1}$) **then**

4: Fail.

5: Let $s_{\mathfrak{b}} = \mathcal{O}(\mathfrak{b})$.

6: **Return** $\underbrace{s_{\mathfrak{b}}}_{\text{small}} \cdot \underbrace{y}_{\text{small}}$.

Our $\mathcal{P}^{-1}$-ideal-SVP to $\mathcal{P}$-ideal-SVP reduction

The oracle $\mathcal{O}$ solves $\mathrm{ideal\text{-}HSVP}$ for $\mathfrak{p}$ uniform prime of norm in $[A, B]$.

Algorithm 4.2 Outline of the $\mathcal{P}^{-1}$-ideal-SVP to $\mathcal{P}$-ideal-SVP reduction

---

Input: An ideal $I = \mathfrak{p}^{-1}$ with $\mathfrak{p}$ uniform prime of norm in $[A, B]$.

Output: $x \in I \setminus \{0\}$ small.

1: Let $s_{\mathfrak{p}} = \mathcal{O}(\mathfrak{p})$. ($\mathfrak{p}$ is uniform)

2: Let $(\mathfrak{b}, y) = \texttt{SampleIdeal}_{A,B}(\mathfrak{p}, s_{\mathfrak{p}})$

3: **if** $\mathfrak{b}$ is not prime. (with probability $(\mathrm{poly} \cdot \rho_K)^{-1}$) **then**

4: Fail.

5: Let $s_{\mathfrak{b}} = \mathcal{O}(\mathfrak{b})$.

6: **Return** $\underbrace{s_{\mathfrak{b}}}_{\text{small}} \cdot \underbrace{y}_{\text{small}}$.

A note on rejection Sampling

We fail if $\mathfrak{b}$ is not prime: we have to do rejection sampling.
The expected number of rejection is

$$\frac{|\mathcal{I}_{A,B}|}{|\mathcal{P}_{A,B}|} \approx \rho_K = \mathrm{Res}_{s=1}\, \zeta_K(s).$$

This quantity can be exponential for some fields (E.g., multiquadratics).

A note on rejection Sampling

We fail if $\mathfrak{b}$ is not prime: we have to do rejection sampling.
The expected number of rejection is

$$\frac{|\mathcal{I}_{A,B}|}{|\mathcal{P}_{A,B}|} \approx \rho_K = \operatorname{Res}_{s=1} \zeta_K(s).$$

This quantity can be exponential for some fields (E.g., multiquadratics).

Also, we lack good approximations for small A, B.

Application to NTRU

NTRU cryptosystem

Proposed first in [HPS96]. In NIST's post-quantum standardization process: **NTRU** and **NTRUPrime**.

Let q be an integer.

Definition ($NTRU_q$)

Let $f, g \in \mathcal{O}_K$ with coefficients $\ll \sqrt{q}$ and f invertible mod q.
Given $h \in \mathcal{O}_K$ such that $f \cdot h = g \bmod q$, find a small multiple of (f, g).

Advantages:

- Small keys.
- Fast encryption/decryption (much faster than RSA).
- Old.

[HPS96]: J. Hoffstein, J. Pipher, J. Silverman. ANTS 1998.

NTRU instances from ideal-HSVP [PS21]

Karp reduction from [PS21].

Ideal SVP	**NTRU**
$\mathfrak{a} = (z) \bigcap \mathcal{O}_K$.	$q \approx V^{2/d}$.
$\mathrm{Vol}(\mathfrak{a}) = V$.	$h = \lfloor q/z \rceil$.
$\mathrm{SVP}(\mathfrak{a}) = s$	$(g, f) = (s, s \cdot \{q/z\})$

NTRU instances from ideal-HSVP [PS21]

Karp reduction from [PS21].

Ideal SVP	**NTRU**
$\mathfrak{a} = (z) \bigcap \mathcal{O}_K$.	$q \approx V^{2/d}$.
$\mathrm{Vol}(\mathfrak{a}) = V$.	$h = \lfloor q/z \rceil$.
$\mathrm{SVP}(\mathfrak{a}) = s$	$(g, f) = (s, s \cdot \{q/z\})$

Distribution of NTRU instances (D^{NTRU}): sample $\mathfrak{p}$ uniform small prime and apply the reduction.

Consequence: worst-case based distribution for NTRU

$$\mathrm{NTRU} \text{ for } D^{\mathrm{NTRU}} \geq \mathcal{P}\text{-ideal-SVP} \geq \text{wc-ideal-SVP}.$$

Wrapping up

Contributions and open problems

Contributions:

- We show that solving $\mathrm{ideal\text{-}HSVP}$ on average over inverse of primes is as hard as solving $\mathrm{ideal\text{-}HSVP}$ on average over primes.
- The new reduction gives an NTRU distribution based on a worst-case problem for polynomial modulus.

Contributions and open problems

Contributions:

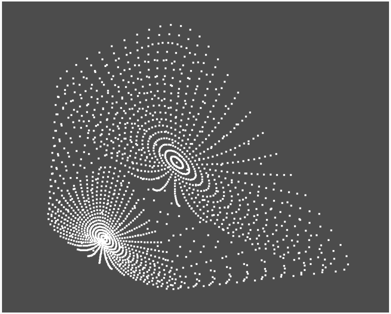
- We show that solving $\mathrm{ideal\text{-}HSVP}$ on average over inverse of primes is as hard as solving $\mathrm{ideal\text{-}HSVP}$ on average over primes.
- The new reduction gives an NTRU distribution based on a worst-case problem for polynomial modulus.

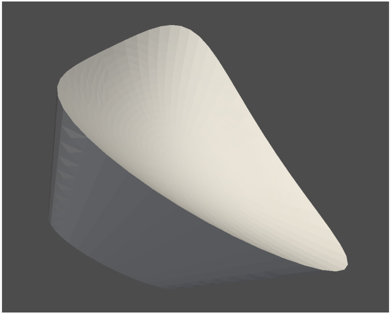
Open problems:

- Can we have such reduction without factoring?
- Can we get rid of the cost in ρ_K?
- Can we have more precise approximates for $|\mathcal{I}_{A,B}|/|\mathcal{P}_{A,B}|$?

Thank you for your attention

Any question?





References i

- K. de Boer, L. Ducas, A. Pellet-Mary, and B. Wesolowski, *Random self-reducibility of Ideal-SVP via Arakelov random walks*, CRYPTO, 2020.
- K. Boudgoust, E. Gachon, and A. Pellet-Mary, *Some easy instances of Ideal-SVP and implications on the partial Vandermonde knapsack problem*, CRYPTO, 2022.
- K. de Boer, *Random walks on arakelov class groups.*, Ph.D. thesis, Leiden University, 2022, Available on request from the author.
- R. Cramer, L. Ducas, C. Peikert, and O. Regev, *Recovering short generators of principal ideals in cyclotomic rings*, EUROCRYPT 2016, 2016.
- R. Cramer, L. Ducas, and B. Wesolowski, *Short Stickelberger class relations and application to Ideal-SVP*, EUROCRYPT, 2017.

References ii

- C. Gentry, *A fully homomorphic encryption scheme*, Ph.D. thesis, Stanford University, 2009.
- A. Pellet-Mary, G. Hanrot, and D. Stehlé, *Approx-SVP in ideal lattices with pre-processing*, EUROCRYPT, 2019.
- A. Pellet-Mary and D. Stehlé, *On the hardness of the NTRU problem*, ASIACRYPT, 2021.
- Quartl, *Matrix pattern qtl3*, 2014, File: `Matrix pattern qtl3.svg`.