Approx-SVP in Ideal lattices with Pre-Processing

Alice Pellet--Mary and Damien Stehlé

LIP, ENS de Lyon

Aric seminar, June 07, 2018
Lattices

A lattice $L$ is a ‘vector space’ over $\mathbb{Z}$.
A lattice $L$ is a ‘vector space’ over $\mathbb{Z}$. A basis of $L$ is an invertible matrix $B$ such that $L = \{Bx \mid x \in \mathbb{Z}^n\}$.

\[
\begin{pmatrix} 3 & 1 \\ 0 & 2 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 17 & 10 \\ 4 & 2 \end{pmatrix}
\]
are two basis of the above lattice.
Lattices

Shortest Vector Problem (SVP)

Find a shortest (in Euclidean norm) non-zero vector. Its Euclidean norm is denoted $\lambda_1$. 
Approximate Shortest Vector Problem (approx-SVP)

Find a short (in Euclidean norm) non-zero vector. (of norm $\leq 2\lambda_1$ for instance).
Closest Vector Problem (CVP)

Given a target point $t$, find a point of the lattice closest to $t$. 

Lattices
Approximate Closest Vector Problem (approx-CVP)

Given a target point $t$, find a point of the lattice close to $t$. 
Complexity of SVP/CVP

Applications

SVP and CVP in general lattices are conjectured to be hard to solve both quantumly and classically $\Rightarrow$ used in cryptography
Complexity of SVP/CVP

Applications

SVP and CVP in general lattices are conjectured to be hard to solve both quantumly and classically ⇒ used in cryptography

Best Time/Approximation trade-off for general lattices: BKZ algorithm
Structured lattices

Improve efficiency of lattice-based crypto using structured lattices.

- Lattice defined using circulant matrices
- Ideal lattices
- ...
Structured lattices

Improve efficiency of lattice-based crypto using structured lattices.

- Lattice defined using circulant matrices
- Ideal lattices
- ...

**RLWE**

The Ring Learning with Error (RLWE) problem is at least as hard as approx-SVP in ideal lattices.

Many cryptographic constructions based on RLWE.
Structured lattices

Improve efficiency of lattice-based crypto using structured lattices.
- Lattice defined using circulant matrices
- Ideal lattices
- ...

RLWE
The Ring Learning with Error (RLWE) problem is at least as hard as approx-SVP in ideal lattices.

Many cryptographic constructions based on RLWE.

Is approx-SVP still hard when restricted to ideal lattices?
SVP in ideal lattices

[CDPR16,CDW17]: Better than BKZ in the quantum setting

---


This work

- **Heuristic**
- **Pre-processing** $2^{O(n)}$ independent of the choice of the ideal (non-uniform algorithm).
This work

- Heuristic
- Pre-processing $2^{O(n)}$ independent of the choice of the ideal (non-uniform algorithm).

Disclaimer: In this talk, only principal ideal lattices
Outline of the talk

1 Definitions and objective

2 The CDPR algorithm

3 This work
First definitions

Notation

\[ R = \mathbb{Z}[X]/(X^n + 1) \text{ for } n = 2^k \]
First definitions

**Notation**

\[ R = \mathbb{Z}[X]/(X^n + 1) \text{ for } n = 2^k \]

- Units: \( R^\times = \{ a \in R \mid \exists b \in R, ab = 1 \} \)
  - E.g. \( \mathbb{Z}^\times = \{1, -1\} \).
First definitions

Notation

\[ R = \mathbb{Z}[X]/(X^n + 1) \text{ for } n = 2^k \]

- Units: \( R^\times = \{a \in R \mid \exists b \in R, ab = 1\} \)
  - E.g. \( \mathbb{Z}^\times = \{1, -1\} \).

- Principal ideals: \( \langle g \rangle = \{gr \mid r \in R\} \) (i.e. all multiples of \( g \))
  - \( g \) is called a generator of \( \langle g \rangle \)
  - The generators of \( \langle g \rangle \) are exactly the \( ug \) for \( u \in R^\times \)
  - E.g. in \( \mathbb{Z} \): \( \langle 2 \rangle = \{\text{even numbers}\} = \langle -2 \rangle \)
Geometric structure

For all $r \in R$, $r = r_0 + r_1X + \cdots + r_{n-1}X^{n-1}$, with $r_i \in \mathbb{Z}$.

- Euclidean norm: $\|r\| = \sqrt{\sum_{i=0}^{n-1} r_i^2}$.
- $R \cong \mathbb{Z}^n$ is a lattice.
Geometric structure

For all \( r \in R \), \( r = r_0 + r_1 X + \cdots + r_{n-1} X^{n-1} \), with \( r_i \in \mathbb{Z} \).

- Euclidean norm: \( \|r\| = \sqrt{\sum_{i=0}^{n-1} r_i^2} \).
- \( R \cong \mathbb{Z}^n \) is a lattice.
- \( \langle g \rangle \) is a sub-lattice of \( R \).
  - E.g. \( \langle 2 \rangle \cong (2\mathbb{Z})^n \).
Geometric structure

For all \( r \in R \), \( r = r_0 + r_1X + \cdots + r_{n-1}X^{n-1} \), with \( r_i \in \mathbb{Z} \).

- Euclidean norm: \( \|r\| = \sqrt{\sum_{i=0}^{n-1} r_i^2} \).
- \( R \cong \mathbb{Z}^n \) is a lattice.
- \( \langle g \rangle \) is a sub-lattice of \( R \).

Minkowski’s embedding

- \( \zeta \in \mathbb{C} \) primitive \( 2n \)-th root of unity \( (\zeta^{2n} = 1) \)
- \( \sigma(r) = (r(\zeta), r(\zeta^3), \cdots, r(\zeta^{n-1})) \in \mathbb{C}^{n/2} \cong \mathbb{R}^n \)
- \( R \mapsto \sigma(R) \) preserves the geometry (isometry + scaling)
Geometric structure

For all \( r \in R, r = r_0 + r_1X + \cdots + r_{n-1}X^{n-1} \), with \( r_i \in \mathbb{Z} \).

- Euclidean norm: \( \|r\| = \sqrt{\sum_{i=0}^{n-1} r_i^2} \).
- \( R \cong \mathbb{Z}^n \) is a lattice.
- \( \langle g \rangle \) is a sub-lattice of \( R \).

Minkowski’s embedding

- \( \zeta \in \mathbb{C} \) primitive \( 2n \)-th root of unity \( (\zeta^{2n} = 1) \)
- \( \sigma(r) = (r(\zeta), r(\zeta^3), \cdots, r(\zeta^{n-1})) \in \mathbb{C}^{n/2} \cong \mathbb{R}^n \)
- \( R \mapsto \sigma(R) \) preserves the geometry (isometry + scaling)
Algebraic structure

Notation

\[ \sigma(r) = (\tilde{r}_1, \ldots, \tilde{r}_{n/2}) \in \mathbb{C}^{n/2} \]

- **Algebraic norm:** \( \mathcal{N}(r) = \prod_{i=1}^{n/2} |\tilde{r}_i|^2 \in \mathbb{R} \).
  - E.g. in \( \mathbb{R} \): \( \mathcal{N}(2) = 2^n \).
Algebraic structure

Notation

\[ \sigma(r) = \left( \tilde{r}_1, \ldots, \tilde{r}_{n/2} \right) \in \mathbb{C}^{n/2} \]

- Algebraic norm: \( \mathcal{N}(r) = \prod_{i=1}^{n/2} |\tilde{r}_i|^2 \in \mathbb{R} \).
  - E.g. in \( R \): \( \mathcal{N}(2) = 2^n \).

- Properties:
  - \( \mathcal{N}(ab) = \mathcal{N}(a) \cdot \mathcal{N}(b) \) for all \( a, b \in R \),
  - \( \mathcal{N}(a) \geq 1 \) and \( \mathcal{N}(a) \in \mathbb{Z} \) for all \( a \in R \setminus \{0\} \),
  - \( \mathcal{N}(u) = 1 \iff u \in R^\times \).
Relations between algebraic/geometric structures

Reminder: \( \sigma(r) = (\tilde{r}_1, \cdots, \tilde{r}_{n/2}) \)

- \( \|r\| = \sqrt{\sum_i |\tilde{r}_i|^2} \)
- \( \mathcal{N}(r) = \prod_i |\tilde{r}_i|^2 \)
Relations between algebraic/geometric structures

Reminder: \( \sigma(r) = (\tilde{r}_1, \ldots, \tilde{r}_{n/2}) \)

- \( \|r\| = \sqrt{\sum_i |\tilde{r}_i|^2} \)
- \( \mathcal{N}(r) = \prod_i |\tilde{r}_i|^2 \)

- Euclidean/algebraic norm:
  - \( \|r\| \) small \( \Rightarrow \) \( \mathcal{N}(r) \) relatively small.
  - \( \mathcal{N}(r) \) small \( \nRightarrow \) \( \|r\| \) relatively small (e.g. \( (2^{-50}, 2^{50}) \)).
Relations between algebraic/geometric structures

Reminder: \( \sigma(r) = (\tilde{r}_1, \cdots, \tilde{r}_{n/2}) \)

- \( \|r\| = \sqrt{\sum_i |\tilde{r}_i|^2} \)
- \( \mathcal{N}(r) = \prod_i |\tilde{r}_i|^2 \)

- Euclidean/algebraic norm:
  - \( \|r\| \) small \( \Rightarrow \mathcal{N}(r) \) relatively small.
  - \( \mathcal{N}(r) \) small \( \not\Rightarrow \|r\| \) relatively small (e.g. \( (2^{-50}, 2^{50}) \)).

- \( \lambda_1(\langle g \rangle) = \text{poly}(n) \cdot \mathcal{N}(g)^{1/n} \)
Objective of this talk

Objective

Given a basis of a principal ideal \( \langle g \rangle \) and \( \alpha \in (0, 1] \),

Find \( r \in \langle g \rangle \) such that

\[
\|r\| \leq 2^{\tilde{O}(n^\alpha)} \cdot \lambda_1 = 2^{\tilde{O}(n^\alpha)} \cdot \mathcal{N}(g)^{1/n}.
\]
Objective of this talk

Objective

Given a basis of a principal ideal \( \langle g \rangle \) and \( \alpha \in (0, 1] \), find \( r \in \langle g \rangle \) such that \( \|r\| \leq 2^{\tilde{O}(n^\alpha)} \cdot \lambda_1 = 2^{\tilde{O}(n^\alpha)} \cdot \mathcal{N}(g)^{1/n} \).

BKZ algorithm can do it in time \( 2^{\tilde{O}(n^{1-\alpha})} \), can we do better?
Outline of the talk

1. Definitions and objective
2. The CDPR algorithm
3. This work
Overview of the CDPR algorithm (on an idea of [CGS14])

Important points

- Large algebraic norm $\Rightarrow$ large Euclidean norm.
- In $\langle g \rangle$, the elements with the smallest algebraic norm are the generators.

---

[CGS14]: P. Campbell, M. Groves, and D. Shepherd. Soliloquy: A cautionary tale.
[BS16]: J.F. Biasse, F. Song. Efficient quantum algorithms for computing class groups and solving the principal ideal problem in arbitrary degree number fields, SODA.
Overview of the CDPR algorithm (on an idea of [CGS14])

Important points

- Large algebraic norm $\Rightarrow$ large Euclidean norm.
- In $\langle g \rangle$, the elements with the smallest algebraic norm are the generators.

The CDPR algorithm: find a generator with a smallest Euclidean norm

---

[CGS14]: P. Campbell, M. Groves, and D. Shepherd. Soliloquy: A cautionary tale.
[BS16]: J.F. Biasse, F. Song. Efficient quantum algorithms for computing class groups and solving the principal ideal problem in arbitrary degree number fields, SODA.
Overview of the CDPR algorithm (on an idea of [CGS14])

Important points

- Large algebraic norm ⇒ large Euclidean norm.
- In \( \langle g \rangle \), the elements with the smallest algebraic norm are the generators.

The CDPR algorithm: find a generator with a smallest Euclidean norm

- Find a generator \( g_1 \) of \( \langle g \rangle \)
  - [BS16]: quantum time \( \text{poly}(n) \)
  - [BEFGK17]: classical time \( 2^{\tilde{O}(\sqrt{n})} \)

- Find \( u \in R^\times \) which minimizes \( \|ug_1\| \).

---

[CGS14]: P. Campbell, M. Groves, and D. Shepherd. Soliloquy: A cautionary tale.
[BS16]: J.F. Biasse, F. Song. Efficient quantum algorithms for computing class groups and solving the principal ideal problem in arbitrary degree number fields, SODA.
The Log unit lattice

**Definitions**

\[
\text{Log} : \sigma(R) \rightarrow \mathbb{R}^{n/2} \\
(\tilde{r}_1, \cdots, \tilde{r}_{n/2}) \mapsto (\log |\tilde{r}_1|, \cdots, \log |\tilde{r}_{n/2}|)
\]

Let \( \mathbf{1} = (1, \cdots, 1) \) and \( H = \mathbf{1}^{\perp}. \)
The Log unit lattice

Definitions

\[ \text{Log} : \sigma(R) \rightarrow \mathbb{R}^{n/2} \]
\[ (\tilde{r}_1, \cdots, \tilde{r}_{n/2}) \mapsto (\log |\tilde{r}_1|, \cdots, \log |\tilde{r}_{n/2}|) \]

Let \( \mathbf{1} = (1, \cdots, 1) \) and \( H = \mathbf{1} \perp \).

Theorem (Dirichlet)

\[ \Lambda := \text{Log}(R^\times) \text{ is a lattice included in } H. \]
The Log unit lattice

Definitions

\[ \text{Log} : \sigma(R) \to \mathbb{R}^{n/2} \]

\[(\tilde{r}_1, \cdots, \tilde{r}_{n/2}) \mapsto (\log |\tilde{r}_1|, \cdots, \log |\tilde{r}_{n/2}|) \]

Let \( \mathbf{1} = (1, \cdots, 1) \) and \( H = \mathbf{1}^\perp \).

Theorem (Dirichlet)

\( \Lambda := \text{Log}(R^\times) \) is a lattice included in \( H \).

Write \( \text{Log}(r) = h + a\mathbf{1} \), with \( h \in H \)

\[ \|r\| \leq \sqrt{n} \cdot 2^a \cdot 2\|h\| \]

\[ a = \frac{\log |\mathcal{N}(r)|}{n} \]
CDPR (upper bound)

Reminder \((\Log(r) = h + a1)\)
- \(|r| \leq \sqrt{n} \cdot 2^a \cdot 2|h|\)
- \(a = \frac{\log |\N(r)|}{n}\)

The CDPR Algorithm:
- Find a generator \(g_1\) of \(\langle g \rangle\).
  - quantum poly time [BS16]
CDPR (upper bound)

Reminder (Log(r) = h + a1)

- ||r|| ≤ √n · 2^a · 2||h||
- a = \frac{\log |N(r)|}{n}

The CDPR Algorithm:
- Find a generator \( g_1 \) of \( \langle g \rangle \).
  - quantum poly time [BS16]
CDPR (upper bound)

Reminder \((\log(r) = h + a1)\)
- \(\|r\| \leq \sqrt{n} \cdot 2^a \cdot 2\|h\|\)
- \(a = \frac{\log |\mathcal{N}(r)|}{n}\)

The CDPR Algorithm:
- Find a generator \(g_1\) of \(\langle g \rangle\).
  - quantum poly time [BS16]
CDPR (upper bound)

Reminder \((\log(r) = h + a1)\)

- \(\|r\| \leq \sqrt{n} \cdot 2^a \cdot 2\|h\|\)
- \(a = \frac{\log |N(r)|}{n}\)

The CDPR Algorithm:
- Find a generator \(g_1\) of \(\langle g \rangle\).
  - quantum poly time [BS16]
CDPR (upper bound)

Reminder \((\log(r) = h + a1)\)
- \(\|r\| \leq \sqrt{n} \cdot 2^a \cdot 2\|h\|\)
- \(a = \frac{\log |N(r)|}{n}\)

The CDPR Algorithm:
- Find a generator \(g_1\) of \(\langle g \rangle\).
  - quantum poly time [BS16]
CDPR (upper bound)

Reminder \((\log(r) = h + a\mathbf{1})\)

- \(|r| \leq \sqrt{n} \cdot 2^a \cdot 2|h|
- \(a = \frac{\log |N(r)|}{n}\)

The CDPR Algorithm:

- Find a generator \(g_1\) of \(\langle g \rangle\).
  - quantum poly time [BS16]
- Solve CVP in \(\Lambda\).
CDPR (upper bound)

Reminder \( \log(r) = h + a1 \)
- \( \|r\| \leq \sqrt{n} \cdot 2^a \cdot 2^h \)
- \( a = \frac{\log |N(r)|}{n} \)

The CDPR Algorithm:
- Find a generator \( g_1 \) of \( \langle g \rangle \).
  - quantum poly time [BS16]
- Solve CVP in \( \Lambda \).
CDPR (upper bound)

Reminder \((\text{Log}(r) = h + a1)\)

- \(\|r\| \leq \sqrt{n} \cdot 2^a \cdot 2\|h\|\)
- \(a = \frac{\log |\mathcal{N}(r)|}{n}\)

The CDPR Algorithm:

- Find a generator \(g_1\) of \(\langle g \rangle\).
  - quantum poly time [BS16]
- Solve CVP in \(\Lambda\).
  - Good basis of \(\Lambda\)
    - \(\Rightarrow\) CVP in poly time
    - \(\|h\| \leq \tilde{O}(\sqrt{n})\)
CDPR (upper bound)

Reminder \((\log(r) = h + a1)\)
- \(\|r\| \leq \sqrt{n} \cdot 2^a \cdot 2\|h\|\)
- \(a = \frac{\log |\mathcal{N}(r)|}{n}\)

The CDPR Algorithm:
- Find a generator \(g_1\) of \(\langle g \rangle\).
  - quantum poly time [BS16]
- Solve CVP in \(\Lambda\).
  - Good basis of \(\Lambda\)
    \(\Rightarrow\) CVP in poly time
    \(\Rightarrow\) \(\|h\| \leq \tilde{O}(\sqrt{n})\)
- \(\|ug_1\| \leq \mathcal{N}(ug_1)^{1/n} \cdot 2\tilde{O}(\sqrt{n})\)
CDPR (upper bound)

Reminder \((\log(r) = h + a1)\)

- \(\|r\| \leq \sqrt{n} \cdot 2^a \cdot 2^{\|h\|}\)
- \(a = \frac{\log |\mathcal{N}(r)|}{n}\)
- \(\lambda_1 = \text{poly}(n) \cdot \mathcal{N}(g)^{1/n}\)

The CDPR Algorithm:

- Find a generator \(g_1\) of \(\langle g \rangle\).
  - quantum poly time [BS16]
- Solve CVP in \(\Lambda\).
  - Good basis of \(\Lambda\)
    \(\Rightarrow\) CVP in poly time
    \(\Rightarrow\) \(\|h\| \leq \tilde{O}(\sqrt{n})\)

\[\|ug_1\| \leq \mathcal{N}(ug_1)^{1/n} \cdot 2\tilde{O}(\sqrt{n}) \leq 2\tilde{O}(\sqrt{n}) \cdot \lambda_1\]
CDPR (upper bound)

Reminder ($\log(r) = h + a1$)

- $\|r\| \leq \sqrt{n} \cdot 2^a \cdot 2\|h\|$
- $a = \frac{\log |N(r)|}{n}$
- $\lambda_1 = \text{poly}(n) \cdot \mathcal{N}(g)^{1/n}$

The CDPR Algorithm:

- Find a generator $g_1$ of $\langle g \rangle$.
  - quantum poly time [BS16]
- Solve CVP in $\Lambda$.
  - Good basis of $\Lambda$
    - $\Rightarrow$ CVP in poly time
    - $\Rightarrow \|h\| \leq \tilde{O}(\sqrt{n})$

\[ \|ug_1\| \leq \mathcal{N}(ug_1)^{1/n} \cdot 2\tilde{O}(\sqrt{n}) \]
\[ \leq 2\tilde{O}(\sqrt{n}) \cdot \lambda_1 \]
CDPR (lower bound)

Reminder (Log(r) = h + a1):
- $\|r\| \leq \sqrt{n} \cdot 2^a \cdot 2\|h\|$
- $a = \frac{\log |\mathcal{N}(r)|}{n}$
- $\lambda_1 = \text{poly}(n) \cdot \mathcal{N}(g)^{1/n}$

Lower bound [CDPR16]:
There exists $t \in H$ such that

$$\forall u \in \mathbb{R}^\times, \|t - \text{Log}(u)\| \geq \Omega(\sqrt{n}).$$
CDPR (lower bound)

Reminder (Log($r$) = $h + a1$)

- $\|r\| \leq \sqrt{n} \cdot 2^a \cdot 2^h$
- $a = \frac{\log |N(r)|}{n}$
- $\lambda_1 = \text{poly}(n) \cdot \mathcal{N}(g)^{1/n}$

Lower bound [CDPR16]:
There exists $t \in H$ such that

$$\forall u \in R^\times, \|t - \text{Log}(u)\| \geq \Omega(\sqrt{n}).$$
CDPR (lower bound)

Reminder: \( \log(r) = h + a1 \)

- \( \|r\| \leq \sqrt{n} \cdot 2^a \cdot 2^h \)
- \( a = \frac{\log |N(r)|}{n} \)
- \( \lambda_1 = \text{poly}(n) \cdot N(g)^{1/n} \)

Lower bound [CDPR16]:
There exists \( t \in H \) such that

\[ \forall u \in \mathbb{R}^\times, \|t - \log(u)\| \geq \Omega(\sqrt{n}). \]
CDPR (lower bound)

Reminder \((\log(r) = h + a1)\):

- \(\|r\| \leq \sqrt{n} \cdot 2^a \cdot 2\|h\|\)
- \(a = \frac{\log |N(r)|}{n}\)
- \(\lambda_1 = \text{poly}(n) \cdot \mathcal{N}(g)^{1/n}\)

**Lower bound [CDPR16]:**

There exists \(t \in H\) such that

\[
\forall u \in R^\times, \|t - \log(u)\| \geq \Omega(\sqrt{n}).
\]
CDPR (lower bound)

Reminder \((\log(r) = h + a1)\)

- \(|r| \leq \sqrt{n} \cdot 2^a \cdot 2^{|h|}\)
- \(a = \frac{\log{|\mathcal{N}(r)|}}{n}\)
- \(\lambda_1 = \text{poly}(n) \cdot \mathcal{N}(g)^{1/n}\)

Lower bound [CDPR16]:
There exists \(t \in H\) such that

\[
\forall u \in R^\times, \|t - \log(u)\| \geq \Omega(\sqrt{n}).
\]

\(\exists \langle g \rangle\) such that, \(\forall u \in R^\times\)

\[
\|ug\| \geq 2^{\Omega(\sqrt{n})} \cdot \lambda_1
\]
Outline of the talk

1. Definitions and objective
2. The CDPR algorithm
3. This work
Idea
Idea
Idea
Idea

\[ \text{Log}(g_1) \]

\[ 2\text{Log}(r) \]

\[ \text{Log}(r) \]

\[ -\text{Log}(r) \]

\[ H \]

\[ t \]

\[ \Lambda \]
Idea
Idea

\[ \log(g_1) \]

\[ 2\log(r) \]

\[ \log(r) \]

\[ H \]

\[ t \]

\[ \Lambda \]
Idea
Idea
Formalisation

**Difficulties**

- We cannot subtract $\log(r_i)$
- We cannot add too many $\log(r_i)$'s

$\Rightarrow$ This is not a lattice
Formalisation

**Difficulties**
- We cannot subtract $\log(r_i)$
- We cannot add too many $\log(r_i)$’s

⇒ This is not a lattice

We consider the lattice

<table>
<thead>
<tr>
<th>$\Lambda$</th>
<th>$h_{\log r_1}, \ldots, h_{\log r_n}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>⋮</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

![Diagram](image)
Formalisation

Difficulties

- We cannot subtract $\log(r_i)$
- We cannot add too many $\log(r_i)$’s

⇒ This is not a lattice

We consider the lattice and CVP target

\[
\begin{array}{c|c}
\Lambda & h_{\log r_1}, \ldots, h_{\log r_n} \\
\hline
0 & 1 \quad 1 \\
& \ddots \\
& 1
\end{array}
\]

\[
\begin{array}{c|c}
\Lambda & -h_{\log g_1} \\
\hline
0 & 0
\end{array}
\]
Formalisation

**Difficulties**

- We cannot subtract $\log(r_i)$
- We cannot add too many $\log(r_i)$’s

$\Rightarrow$ This is not a lattice

We consider the lattice and CVP target

<table>
<thead>
<tr>
<th>$\Lambda$</th>
<th>$h_{\log r_1}, \ldots, h_{\log r_n}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$1$</td>
</tr>
<tr>
<td></td>
<td>$1$</td>
</tr>
<tr>
<td></td>
<td>$\ddots$</td>
</tr>
<tr>
<td></td>
<td>$1$</td>
</tr>
</tbody>
</table>

$-h_{\log g_1}$

$c > 0$
Compute $r_1, \ldots, r_n$ of small algebraic norms $p(n)/2 \sim O(\sqrt{n})$

generate $\langle g \rangle$ polynomial $\langle n \rangle/2 \sim O(\sqrt{n})$

$\Lambda_0 h \log r_1, \ldots, h \log r_n$

Construct $L$ and $t = -h \log g_1$

Solve CVP in $L$ with target $t$ (for some $\alpha \in [0, 1]$)

$\Rightarrow$ get a vector $s \in L$ such that $\|s - t\| \leq \tilde{O}(n^{\alpha})$

Write $s = h \log r^{\star}$ for some $r \in \mathbb{R}$ polynomial $\langle n \rangle$

$\|rg_1\| \leq 2 \tilde{O}(n^{\alpha}) \cdot \lambda_1$
Summary

Compute $r_1, \cdots, r_n$ of small algebraic norms
Summary

Compute $r_1, \ldots, r_n$ of small algebraic norms

Compute $g_1$ a generator of $\langle g \rangle$
Summary

Compute $r_1, \ldots, r_n$ of small algebraic norms

Compute $g_1$ a generator of $\langle g \rangle$

Construct $L := \begin{pmatrix} \Lambda & h_{\log r_1}, \ldots, h_{\log r_n} \\ 0 & 1 & 1 & \cdots & 1 \end{pmatrix}$ and $t := \begin{pmatrix} -h_{\log g_1} \\ c > 0 \end{pmatrix}$
Summary

Compute $r_1, \cdots, r_n$ of small algebraic norms

Compute $g_1$ a generator of $\langle g \rangle$

Construct $L := \begin{pmatrix} \Lambda & h_{\log r_1}, \ldots, h_{\log r_n} \\ 0 & 1 & 1 & \cdots & 1 \end{pmatrix}$ and $t := \begin{pmatrix} -h_{\log g_1} \\ c > 0 \end{pmatrix}$

Solve CVP in $L$ with target $t$ (for some $\alpha \in [0, 1]$)

$\Rightarrow$ get a vector $s \in L$ such that $\|s - t\| \leq \tilde{O}(n^\alpha)$
Summary

Compute \( r_1, \ldots, r_n \) of small algebraic norms

Compute \( g_1 \) a generator of \( \langle g \rangle \)

Construct \( L := \begin{array}{c|c}
\Lambda & h_{\log r_1}, \ldots, h_{\log r_n} \\
0 & 1 \\
& 1 \\
& \ddots \\
& 1 \\
\end{array} \) and \( t := \begin{array}{c}
-h_{\log g_1} \\
c > 0
\end{array} \)

Solve CVP in \( L \) with target \( t \) (for some \( \alpha \in [0, 1] \))
\( \Rightarrow \) get a vector \( s \in L \) such that \( \|s - t\| \leq \tilde{O}(n^\alpha) \)

Write \( s = \begin{array}{c}
h_{\log r} \\
\ast
\end{array} \) for some \( r \in R \)
Summary

Compute $r_1, \cdots, r_n$ of small algebraic norms

Compute $g_1$ a generator of $\langle g \rangle$

\[
\begin{array}{|c|c|}
\hline
& h_{\log r_1}, \ldots, h_{\log r_n} \\
\hline
\Lambda & 1 \\
0 & 1 \\
& \ddots \\
& 1 \\
\hline
\end{array}
\]

Construct $L := \Lambda$ and $t := \begin{bmatrix} -h_{\log g_1} \\ c > 0 \end{bmatrix}$

Solve CVP in $L$ with target $t$ (for some $\alpha \in [0, 1]$)

⇒ get a vector $s \in L$ such that $\|s - t\| \leq \tilde{O}(n^\alpha)$

Write $s = \begin{bmatrix} h_{\log r} \\ \ast \end{bmatrix}$ for some $r \in R$

\[\|rg_1\| \leq 2\tilde{O}(n^\alpha) \cdot \lambda_1\]
Summary

Compute \( r_1, \cdots, r_n \) of small algebraic norms

Compute \( g_1 \) a generator of \( \langle g \rangle \)

\[
\begin{array}{c|cccc}
\Lambda & h_{\log r_1}, & \ldots, & h_{\log r_n} \\
\hline
0 & 1 & 1 & \cdots \\
1 & 1 & \cdots \\
\end{array}
\]

Construct \( L := \) and \( t := \)

Solve CVP in \( L \) with target \( t \) (for some \( \alpha \in [0, 1] \))

\[ \Rightarrow \text{get a vector } s \in L \text{ such that } \| s - t \| \leq \tilde{O}(n^\alpha) \]

Write \( s = h_{\log r} \) for some \( r \in R \)

\[ \|rg_1\| \leq 2\tilde{O}(n^\alpha) \cdot \lambda_1 \]
Summary

Compute $r_1, \cdots, r_n$ of small algebraic norms \(\text{poly}(n) / 2^{\tilde{O}(\sqrt{n})}\)

Compute $g_1$ a generator of \(\langle g \rangle\) \(\text{poly}(n) / 2^{\tilde{O}(\sqrt{n})}\)

\[
\begin{array}{c|c|c|c}
\Lambda & h_{\log r_1}, & \cdots, & h_{\log r_n} \\
0 & 1 & 1 & \cdots \\
& 1 \\
\end{array}
\]

Construct \(L := \) and \(t := \)

\[
\begin{pmatrix}
-h_{\log g_1} \\
c > 0
\end{pmatrix}
\]

Solve CVP in $L$ with target $t$ (for some $\alpha \in [0, 1]$)

\(\Rightarrow\) get a vector $s \in L$ such that \(\|s - t\| \leq \tilde{O}(n^\alpha)\)

Write \(s = \)

\[
\begin{pmatrix}
h_{\log r} \\
\ast \\
\end{pmatrix}
\]

for some \(r \in R\) \(\text{poly}(n)\)

\[
\|rg_1\| \leq 2^{\tilde{O}(n^\alpha)} \cdot \lambda_1
\]
Summary

Compute $r_1, \cdots, r_n$ of small algebraic norms

Compute $g_1$ a generator of $\langle g \rangle$

\[
\begin{array}{c|c}
\Lambda & h_{\log r_1}, \ldots, h_{\log r_n} \\
\hline
0 & 1 \ 1 \\
1 & \cdots \\
1 & \\
\end{array}
\quad \text{and} \quad t := \begin{bmatrix} -h_{\log g_1} \\ c > 0 \end{bmatrix}
\]

Construct $L := \Lambda$ and $t := \begin{bmatrix} -h_{\log g_1} \\ c > 0 \end{bmatrix}$

Solve CVP in $L$ with target $t$ (for some $\alpha \in [0, 1]$)

$\Rightarrow$ get a vector $s \in L$ such that $\|s - t\| \leq \tilde{O}(n^\alpha)$

Write $s = \begin{bmatrix} h_{\log r} \\ \ast \end{bmatrix}$ for some $r \in R$

\[\|rg_1\| \leq 2\tilde{O}(n^\alpha) \cdot \lambda_1\]
How to solve CVP in $L$?

<table>
<thead>
<tr>
<th>CDPR</th>
<th>This work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Good basis of $\Lambda$</td>
<td>No good basis of $L$ known</td>
</tr>
</tbody>
</table>

How to solve CVP in $L$?

<table>
<thead>
<tr>
<th></th>
<th>CDPR</th>
<th>This work</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Good basis of $\Lambda$</td>
<td>No good basis of $L$ known</td>
</tr>
</tbody>
</table>

**Key observation**

$$L := \begin{pmatrix}
\Lambda & h_{\log n}, \ldots, h_{\log n} \\
0 & 1 & 1 \\
& & \ddots \\
& & & 1
\end{pmatrix}$$

does not depend on $\langle g \rangle$

---

How to solve CVP in $L$?

<table>
<thead>
<tr>
<th>CDPR</th>
<th>This work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Good basis of $\Lambda$</td>
<td>No good basis of $L$ known</td>
</tr>
</tbody>
</table>

Key observation

$L := \begin{pmatrix} \Lambda & h_{\log n}, \ldots, h_{\log n} \\ 0 & 1 \\ & \ddots \\ & & 1 \end{pmatrix}$ does not depend on $\langle g \rangle \Rightarrow$ Pre-processing on $L$

---


A. Pellet--Mary

Approx-SVP in Ideal lattices

Aric seminar
How to solve CVP in $L$?

<table>
<thead>
<tr>
<th>CDPR</th>
<th>This work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Good basis of $\Lambda$</td>
<td>No good basis of $L$ known</td>
</tr>
</tbody>
</table>

**Key observation**

$$L := \begin{bmatrix} \Lambda & h_{\log r_1}, \ldots, h_{\log r_n} \\ 0 & 1 & 1 & \cdots & 1 \end{bmatrix}$$

does not depend on $\langle g \rangle \Rightarrow \text{Pre-processing on } L$

[Laa16]:  
- Find $s \in L$ such that $\|s - t\| = \tilde{O}(n^\alpha)$
- Time: $2\tilde{O}(n^{1-2\alpha})$ (query)  
  + $2^{O(n)}$ (pre-processing)

---

Conclusion

<table>
<thead>
<tr>
<th>Approximation</th>
<th>Query time</th>
<th>Pre-processing</th>
</tr>
</thead>
<tbody>
<tr>
<td>(2\tilde{O}(n^\alpha))</td>
<td>(2\tilde{O}(n^{1-2\alpha}) + (\text{poly}(n) \text{ or } 2\tilde{O}(\sqrt{n})))</td>
<td>(2^O(n))</td>
</tr>
</tbody>
</table>

\[ +2^O(n) \text{ Pre-processing} / \text{Non-uniform algorithm} \]
Open problems

- Generalization to other number fields?
- Removing (or testing) the heuristics
Open problems

- Generalization to other number fields?
- Removing (or testing) the heuristics

Questions?