Approx-SVP in Ideal lattices with Pre-Processing

Alice Pellet-Mary, Guillaume Hanrot and Damien Stehlé

LIP, ENS de Lyon

Journées C2 2018, October 8
What is this talk about

Time/Approximation factor trade-off for SVP in ideal lattices:

- **BKZ algorithm**
- [CDPR16,CDW17]
- This work (with $2^{O(n)}$ pre-processing)
A lattice $L$ is a discrete ‘vector space’ over $\mathbb{Z}$. 

\[
\begin{pmatrix}
3 & 1 \\
0 & 2
\end{pmatrix}
\] 

and 

\[
\begin{pmatrix}
17 & 11 \\
4 & 2
\end{pmatrix}
\] 

are two bases of the above lattice.
A lattice \( L \) is a discrete ‘vector space’ over \( \mathbb{Z} \).
A basis of \( L \) is an invertible matrix \( B \) such that \( L = \{Bx | x \in \mathbb{Z}^n\} \).

\[
\begin{pmatrix} 3 & 1 \\ 0 & 2 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 17 & 11 \\ 4 & 2 \end{pmatrix}
\] are two bases of the above lattice.
Shortest Vector Problem (SVP)

Find a shortest (in Euclidean norm) non-zero vector. Its Euclidean norm is denoted $\lambda_1$. 

Lattices
Approximate Shortest Vector Problem (approx-SVP)

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

Given a target point $t$, find a point of the lattice closest to $t$. 
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 ⇒ used in cryptography

<table>
<thead>
<tr>
<th>Best Time/Approximation trade-off for general lattices: BKZ algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
</tr>
<tr>
<td>$2^n$</td>
</tr>
</tbody>
</table>

A. Pellet-Mary

Approx-SVP in Ideal lattices

JC2 2018
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

![Graph showing the trade-off between time and approximation factor](graph.png)
Structured lattices

Improve efficiency of lattice-based crypto using structured lattices.
⇒ E.g. ideal lattices
Structured lattices

Improve efficiency of lattice-based crypto using structured lattices.
⇒ E.g. ideal lattices

Is \textit{approx-SVP} still hard when restricted to ideal lattices?
SVP in ideal lattices

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

- Heuristic
- For prime power cyclotomic fields

\[ \text{poly} \rightarrow 2^n \rightarrow 2^{n^{0.5}} \rightarrow \text{poly} \]

\[ \text{Time} \rightarrow 2^n \rightarrow 2^{n^{0.5}} \rightarrow \text{Approximation factor} \]

---


This work

- Heuristic
- Pre-processing $2^{O(n)}$, independent of the choice of the ideal (non-uniform algorithm).
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 \( g \))
  - e.g. \( \langle 2 \rangle = \{ \text{even numbers} \} \) in \( \mathbb{Z} \)
  - \( g \) is called a generator of \( \langle g \rangle \)
  - The generators of \( \langle g \rangle \) are exactly the \( ug \) for \( u \in R^\times \)
Why is $\langle g \rangle$ a lattice?

\[ R \cong \mathbb{Z}^n \]

\[ R = \mathbb{Z}[X]/(X^n + 1) \rightarrow \mathbb{Z}^n \]

\[ r = r_0 + r_1X + \cdots + r_{n-1}X^{n-1} \mapsto (r_0, r_1, \ldots, r_{n-1}) \]
Why is $\langle g \rangle$ a lattice?

$R \simeq \mathbb{Z}^n$

\[ R = \mathbb{Z}[X]/(X^n + 1) \rightarrow \mathbb{Z}^n \]
\[ r = r_0 + r_1X + \cdots + r_{n-1}X^{n-1} \mapsto (r_0, r_1, \ldots, r_{n-1}) \]

$\langle g \rangle \subseteq R \simeq \mathbb{Z}^n$ + stable by ‘+’ and ‘-’ $\Rightarrow$ lattice
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 $\|r\| \leq 2^{\tilde{O}(n^\alpha)} \cdot \lambda_1$. 

The BKZ algorithm can do it in time $2^{O\left(n^{1-\alpha}\right)}$, can we do better?

Time Approximation factor $2^{n} \cdot 2^{n/2}$ poly $2^{n} \cdot 2^{n/2}$ poly

A. Pellet-Mary

Approx-SVP in Ideal lattices

JC2 2018
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.

BKZ algorithm can do it in time $2^{O(n^{1-\alpha})}$, can we do better?

![Graph showing the relationship between time and approximation factor]
Outline of the talk

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

**Idea**

Maybe $g$ is a somehow small element of $\langle g \rangle$
Main idea of the CDPR algorithm (on an idea of [CGS14])

Idea

Maybe $g$ is a somehow small element of $\langle g \rangle$

If $n = 1$: e.g. $\langle 2 \rangle \Rightarrow 2$ and $-2$ are the smallest elements.

\[
\begin{array}{cccccc}
-6 & -4 & -2 & 0 & 2 & 4 & 6 \\
\end{array}
\]

[CGS14]: P. Campbell, M. Groves, and D. Shepherd. Soliloquy: A cautionary tale.
Main idea of the CDPR algorithm (on an idea of [CGS14])

Idea

Maybe $g$ is a somehow small element of $\langle g \rangle$

If $n = 1$: e.g. $\langle 2 \rangle \Rightarrow 2$ and $-2$ are the smallest elements.

-6  -4  -2  0  2  4  6

For larger $n$: one of the generators is somehow small

[CGS14]: P. Campbell, M. Groves, and D. Shepherd. Soliloquy: A cautionary tale.
The Log space

Log : $R \rightarrow \mathbb{R}^n$ (somehow generalising log to $R$)

Let $1 = (1, \cdots, 1)$ and $H = 1^\perp$. 

\[ \Lambda := \text{Log}(R \times \mathbb{R}) \text{ is a lattice} \]

\[ \text{Log}(r_1 \cdot r_2) = \text{Log}(r_1) + \text{Log}(r_2) \]

\[ \|r\| \simeq 2^{\|\text{Log}(r)\|_\infty} \]
The Log space

Log : \( R \rightarrow \mathbb{R}^n \) (somehow generalising log to \( R \))

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

**Properties**

\[
\text{Log } r = h + a1, \text{ with } h \in H
\]

- \( a \geq 0 \)
The Log space

Log : $R \rightarrow \mathbb{R}^n$ (somehow generalising log to $R$)

Let $1 = (1, \cdots, 1)$ and $H = 1^\perp$.

**Properties**

$\text{Log } r = h + a1$, with $h \in H$

- $a \geq 0$
- $a = 0$ iff $r$ is a unit
- $\Lambda := \text{Log}(R^\times)$ is a lattice
The Log space

Log : $R \rightarrow \mathbb{R}^n$ (somehow generalising log to $R$)

Let $1 = (1, \cdots, 1)$ and $H = 1^\perp$.

Properties

$\text{Log } r = h + a1$, with $h \in H$

- $a \geq 0$
- $a = 0$ iff $r$ is a unit
- $\Lambda := \text{Log}(R^\times)$ is a lattice
- $\text{Log}(r_1 \cdot r_2) = \text{Log}(r_1) + \text{Log}(r_2)$
The Log space

Log : $R \rightarrow \mathbb{R}^n$ (somehow generalising log to $R$)

Let $1 = (1, \cdots, 1)$ and $H = 1^\perp$.

Properties

Log $r = h + a1$, with $h \in H$

- $a \geq 0$
- $a = 0$ iff $r$ is a unit
- $\Lambda := \text{Log}(R^\times)$ is a lattice
- $\text{Log}(r_1 \cdot r_2) = \text{Log}(r_1) + \text{Log}(r_2)$
- $\|r\| \simeq 2\|\text{Log} r\|_\infty$
The CDPR algorithm

What does $\text{Log}\langle g \rangle$ look like?
The CDPR algorithm

What does $\text{Log}\langle g \rangle$ look like?

\[
\langle g \rangle
\]

Solve CVP in $\Lambda$

$\text{Good basis of } \Lambda \Rightarrow \text{CVP in poly time}$

$\|h\| \leq \tilde{O}(\sqrt{n})$

$\|ug_1\| \leq 2\tilde{O}(\sqrt{n}) \cdot \lambda_1$

[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 CDPR algorithm

What does $\text{Log}\langle g \rangle$ look like?

---

$\text{Log}(g) + \Lambda$

$H$

$\langle g \rangle$

$\text{Log}(g)$
The CDPR algorithm

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

---

[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 CDPR Algorithm:

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

---

[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 CDPR Algorithm:

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

---

[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 CDPR algorithm

The CDPR Algorithm:

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

---

[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 CDPR algorithm

The CDPR Algorithm:
- Find a generator $g_1$ of $\langle g \rangle$.
  - $[\text{BS16}]:$ quantum time $\text{poly}(n)$
  - $[\text{BEFGK17}]:$ classical time $2^{\widetilde{O}(\sqrt{n})}$
- Solve CVP in $\Lambda$

---

$[\text{BS16}]:$ 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 CDPR algorithm

The CDPR Algorithm:
- Find a generator $g_1$ of $\langle g \rangle$.
  - [BS16]: quantum time $\text{poly}(n)$
  - [BEFGK17]: classical time $2^{\tilde{O}(\sqrt{n})}$
- Solve CVP in $\Lambda$

---

[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 CDPR algorithm

The CDPR Algorithm:

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

- Solve CVP in $\Lambda$

---

[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 CDPR algorithm

The CDPR Algorithm:

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

- Solve CVP in $\Lambda$
  - Good basis of $\Lambda$
    ⇒ CVP in poly time
    ⇒ $\|h\| \leq \tilde{O}(\sqrt{n})$

---

[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 CDPR Algorithm:

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

- Solve CVP in $\Lambda$
  - Good basis of $\Lambda$
    $\Rightarrow$ CVP in poly time
    $\Rightarrow \|h\| \leq \tilde{O}(\sqrt{n})$

\[ \|ug_1\| \leq 2^{\tilde{O}(\sqrt{n})} \cdot \lambda_1 \]

---

[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.

Outline of the talk

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

\[ \log(g_1) \approx \sqrt{n} \]

\[ \log(g_1) + \Lambda \approx \sqrt{n}/2 \]

\[ \Lambda \]

\[ H \]
Idea

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

\[ H \]

\[ \text{Log}(g_1) + \Lambda \]
Idea

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

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

\[ H \]

\[ \text{Log}(g_1) + \Lambda \]
Idea
Idea
Idea
Idea
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>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

$L = \Lambda \cup \bigcup_i (h_{\log r_i} + \Lambda)$ 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

$L = \Lambda \cup \bigcup_{i} (h_{\log r_i} + \Lambda)$ does not depend on $\langle g \rangle \Rightarrow$ Pre-processing on $L$
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**

$L = \Lambda \cup \bigcup_i (h_{\log r_i} + \Lambda)$ does not depend on $\langle g \rangle \implies$ 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>

\[\text{Time} = 2n, \quad 2n^{0.5}, \quad \text{poly} \]

- quantum
- classical

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

- Non principal ideals ✓
- Generalization to other number fields ✓
- Removing the heuristics ?
Extensions

- Non principal ideals ✓
- Generalization to other number fields ✓
- Removing the heuristics ?

Questions?