Improved (Provable) Algorithms for the Shortest Vector Problem via Bounded Distance Decoding

Divesh Aggarwal

Centre for Quantum Technologies

Yanlin Chen

CWI Centrum Wiskunde & Informatica

Rajendra Kumar

Centre for Quantum Technologies

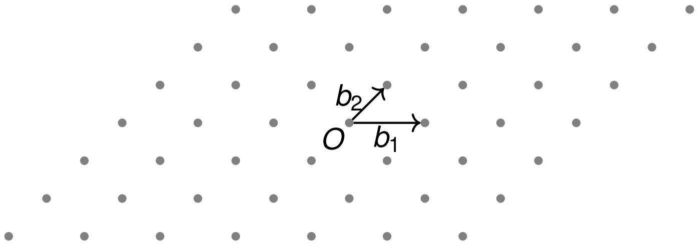
Yixin Shen

Royal Holloway University of London

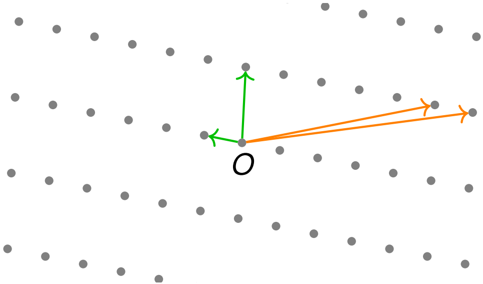
What is a (Euclidean) lattice?

Definition

$\mathcal{L}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n) = \left\{\sum_{i=1}^{n} x_i \boldsymbol{b}_i : x_i \in \mathbb{Z}\right\}$ where $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ is a basis of $\mathbb{R}^n$.

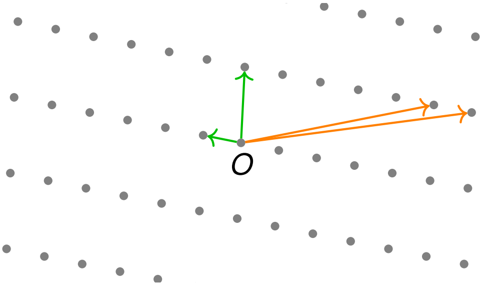


Lattice-based cryptography: fundamental idea



- good basis: private information, makes problem easy
- bad basis: public information, makes problem hard

Lattice-based cryptography: fundamental idea

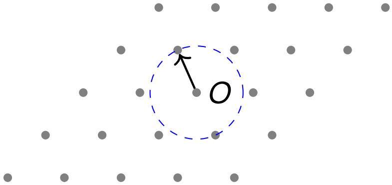


- good basis: private information, makes problem easy
- bad basis: public information, makes problem hard

Basis reduction: transform a bad basis into a good one
Main tool: BKZ algorithm and its variants

Requires to solve the (approx-)SVP problem in smaller dimensions.

The Shortest Vector Problem

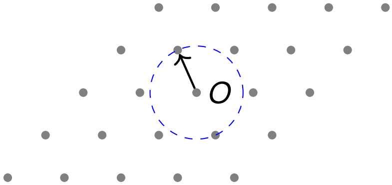


Shortest Vector Problem (SVP):

Given a basis for the lattice $\mathcal{L}$, find a shortest nonzero lattice vector.

$\lambda_1(\mathcal{L})$ = length of such a vector.

The Shortest Vector Problem



Shortest Vector Problem (SVP):
Given a basis for the lattice $\mathcal{L}$, find a shortest nonzero lattice vector.
$\lambda_1(\mathcal{L})$ = length of such a vector.

Main approaches for SVP:

- Enumeration: $2^{O(n\log(n))}$ time and $\mathrm{poly}(n)$ space
- Sieving: $2^{O(n)}$ time and $2^{O(n)}$ space

Sieving

- Heuristic algorithms: fastest in practice
- Provable algorithms: important for theory $\rightarrow$ our work

Results in the Classical Setting

Provable algorithms for SVP:

Time Complexity	Space Complexity	Reference
$n^{\frac{n}{2e}+o(n)}$	poly(n)	[Kan87,HS07]
$2^{n+o(n)}$	$2^{n+o(n)}$	[ADRS15]
$2^{2.05n+o(n)}$	$2^{0.5n+o(n)}$	[CCL18]
$2^{1.7397n+o(n)}$	$2^{0.5n+o(n)}$	Our work

Results in the Classical Setting

Provable algorithms for SVP:

Time Complexity	Space Complexity	Reference
$n^{\frac{n}{2e}+o(n)}$	$\mathrm{poly}(n)$	[Kan87,HS07]
$2^{n+o(n)}$	$2^{n+o(n)}$	[ADRS15]
$2^{2.05n+o(n)}$	$2^{0.5n+o(n)}$	[CCL18]
$2^{1.7397n+o(n)}$	$2^{0.5n+o(n)}$	Our work

Our work: first provable smooth time/space trade-off for SVP

time $q^{13n+o(n)}$ space $\mathrm{poly}(n) \cdot q^{\frac{16n}{q^2}}$ $q \in [4, \sqrt{n}]$

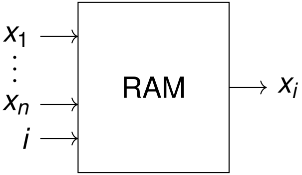
- $q = \sqrt{n}$: time $n^{O(n)}$ and space $\mathrm{poly}(n)$, not as good as [Kan87].
- $q = 4$: time $2^{O(n)}$ and space $2^{O(n)}$, not as good as [ADRS15].

Interlude: quantum memory models

classical access

quantum access

classical data



standard

quantum data

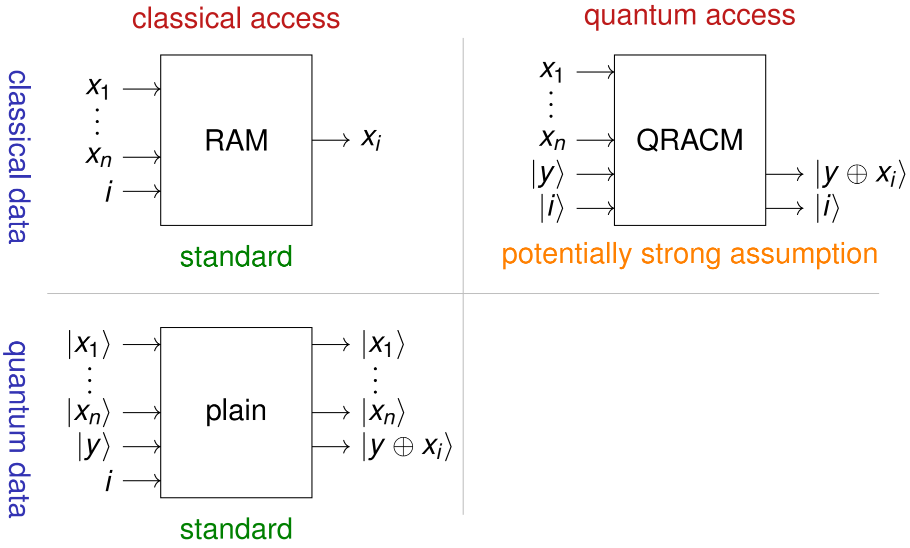
Assumption: $O(1)$ time cost

Interlude: quantum memory models

	classical access	quantum access
classical data	$x_1, \ldots, x_n, i \rightarrow$ RAM $\rightarrow x_i$ standard	
quantum data	$\lvert x_1\rangle, \ldots, \lvert x_n\rangle, \lvert y\rangle, i \rightarrow$ plain $\rightarrow \lvert x_1\rangle, \ldots, \lvert x_n\rangle, \lvert y \oplus x_i\rangle$ standard	

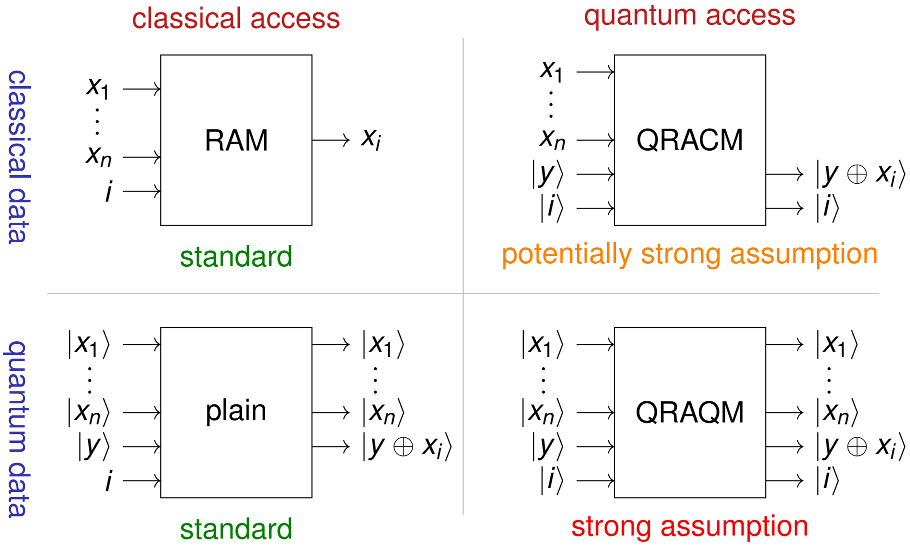
Assumption: $O(1)$ time cost

Interlude: quantum memory models



Assumption: $O(1)$ time cost

Interlude: quantum memory models



Assumption: $O(1)$ time cost

Results in the Quantum Setting

Provable quantum algorithms for SVP:

Time Complexity	Space Complexity			Reference
	Classical	Qubits	Model	
$2^{1.799n+o(n)}$	$2^{1.286n+o(n)}$	poly(n)	QRACM	[LMP15]
$2^{1.2553n+o(n)}$	$2^{0.5n+o(n)}$	poly(n)	plain	[CCL18]
$2^{0.9535n+o(n)}$	$2^{0.5n+o(n)}$	poly(n)	plain	Our work

Remark on quantum heuristic algorithms:

- better complexity: $2^{0.265n+o(n)}$ [Laarhoven15]
- requires QRACM (strong assumption)
- even better complexity: $2^{0.257n+o(n)}$ [CL21]
- requires QRAQM (even stronger assumption)

Sieving Algorithms

Original idea [AKS01]:

- Reduce basis
- Generate random vectors
- Repeat many times:
    - Sieve vectors

Sieving Algorithms

Original idea [AKS01]:

- Reduce basis
- Generate random vectors
- Repeat many times:
  - Sieve vectors

Sieve:

Input: many vectors of length $\leqslant \ell$
Output: many vectors of length $\leqslant \frac{\ell}{2}$

Combine pairs of vectors to produce shorter vectors

Sieving Algorithms

Original idea [AKS01]:

- Reduce basis
- Generate random vectors
- Repeat many times:
  - Sieve vectors

Sieve:

Input: many vectors of length $\leqslant \ell$
Output: many vectors of length $\leqslant \frac{\ell}{2}$

Combine pairs of vectors to produce shorter vectors

Idea: LLL reduced $\rightsquigarrow$ length $\ell \leqslant 2^{O(n)}\lambda_1$, sieve $O(n)$, solve SVP

Sieving Algorithms

Original idea [AKS01]:

- Reduce basis
- Generate random vectors
- Repeat many times:
  - Sieve vectors

Sieve:
Input: many vectors of length $\leqslant \ell$
Output: many vectors of length $\leqslant \frac{\ell}{2}$

Combine pairs of vectors to produce shorter vectors

Idea: LLL reduced $\rightsquigarrow$ length $\ell \leqslant 2^{O(n)}\lambda_1$, sieve $O(n)$, solve SVP

Many heuristic variants: local sensitive hash, tuple sieve, ...
All control the length of the vectors.

Sieving Algorithms

Original idea [AKS01]:

- Reduce basis
- Generate random vectors
- Repeat many times:
  - Sieve vectors

Sieve:
Input: many vectors of length $\leqslant \ell$
Output: many vectors of length $\leqslant \frac{\ell}{2}$

Combine pairs of vectors to produce shorter vectors

Idea: LLL reduced $\rightsquigarrow$ length $\ell \leqslant 2^{O(n)}\lambda_1$, sieve $O(n)$, solve SVP

Many heuristic variants: local sensitive hash, tuple sieve, ...
All control the length of the vectors.

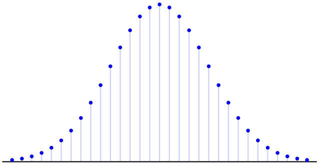
[ADRS15]'s new idea: control distribution instead of length of vectors

Discrete Gaussian Sampling

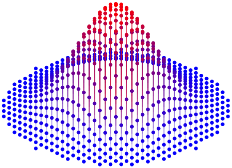
$$\rho_s(\boldsymbol{x}) = \exp\left(-\pi \frac{\|\boldsymbol{x}\|^2}{s^2}\right), \qquad D_{L,s}(\boldsymbol{x}) = \frac{\rho_s(\boldsymbol{x})}{\rho_s(L)}, \qquad \boldsymbol{x} \in \mathbb{R}^n, s > 0.$$

Definition (Discrete Gaussian Distribution)

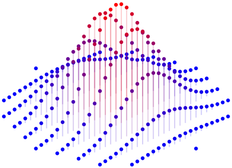
On lattice L with parameter s: probability of $\boldsymbol{x} \in L$ is $D_{L,s}(\boldsymbol{x})$.



$L = \mathbb{Z}, s = 7$



$L = \mathbb{Z}^2, s = 7$



$L = \mathbb{Z} \times 4\mathbb{Z}, s = 7$

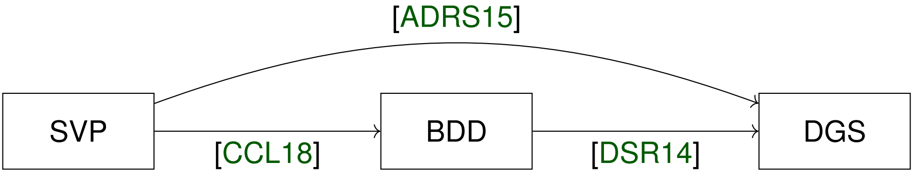
Discrete Gaussian Sampling (DGS)

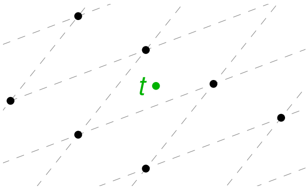
- **input:** L and s
- **output:** random $\boldsymbol{x} \in L$ according to $D_{L,s}$.

DGS, BDD and SVP

$$\begin{tikzcd} \text{SVP} \arrow[rrr, bend left, "\text{[ADRS15]}"] & & & \text{DGS} \end{tikzcd}$$

DGS, BDD and SVP

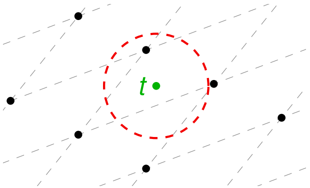




Bounded Distance Decoding ($\alpha-$BDD):
Given a lattice $\mathcal{L}$ and a target vector $t \in \mathbb{R}^n$

DGS, BDD and SVP

$$\begin{tikzcd} \boxed{\text{SVP}} \arrow[r, "\text{[CCL18]}"'] \arrow[rr, bend left, "\text{[ADRS15]}"] & \boxed{\text{BDD}} \arrow[r, "\text{[DSR14]}"'] & \boxed{\text{DGS}} \end{tikzcd}$$

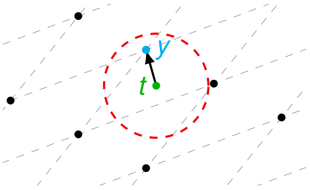


Bounded Distance Decoding ($\alpha-$BDD):
Given a lattice $\mathcal{L}$ and a target vector $t \in \mathbb{R}^n$ with distance to lattice $\leq \alpha \cdot \lambda_1(\mathcal{L})$

The two reductions use completely different DGS parameter regimes!

DGS, BDD and SVP

$$\begin{tikzcd}
\boxed{\text{SVP}} \arrow[r, "\text{[CCL18]}"'] \arrow[rr, bend left, "\text{[ADRS15]}"] & \boxed{\text{BDD}} \arrow[r, "\text{[DSR14]}"'] & \boxed{\text{DGS}}
\end{tikzcd}$$



Bounded Distance Decoding (α−BDD):
Given a lattice $\mathcal{L}$ and a target vector $t \in \mathbb{R}^n$ with distance to lattice $\leq \alpha \cdot \lambda_1(\mathcal{L})$, find the closest vector $y \in \mathcal{L}$.

- α is decoding distance/radius
- $\alpha < \frac{1}{2}$ for unique solution

The two reductions use completely different DGS parameter regimes!

Hardness of Discrete Gaussian Sampling

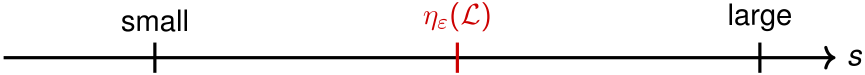
Parameter s (width/standard deviation) of $D_{\mathcal{L},s}$:

- hard to sample
- SVP

- easy to sample

Hardness of Discrete Gaussian Sampling

Parameter s (width/standard deviation) of $D_{\mathcal{L},s}$:



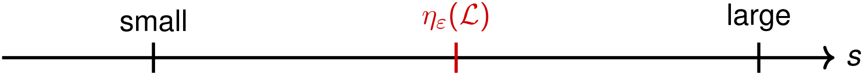
- hard to sample
- SVP

smoothing parameter

- easy to sample

Hardness of Discrete Gaussian Sampling

Parameter s (width/standard deviation) of $D_{\mathcal{L},s}$:



- hard to sample
- SVP

smoothing parameter

- easy to sample

- Open problem: $2^{O(n)}$ time, $2^{o(n)}$ space algorithm for $s = \eta_\varepsilon(\mathcal{L})$
- No known time/space trade-off for $s \ll \eta_\varepsilon(\mathcal{L})$

Hardness of Discrete Gaussian Sampling

Parameter s (width/standard deviation) of $D_{\mathcal{L},s}$:

small | $\eta_\varepsilon(\mathcal{L})$ | large → s

- hard to sample
- SVP

smoothing parameter

- easy to sample

- Open problem: $2^{O(n)}$ time, $2^{o(n)}$ space algorithm for $s = \eta_\varepsilon(\mathcal{L})$
- No known time/space trade-off for $s \ll \eta_\varepsilon(\mathcal{L})$

$$\boxed{\text{SVP}} \xrightarrow{[\text{CCL18}]} \boxed{\text{BDD}} \xrightarrow{[\text{DSR14}]} \boxed{\text{DGS}}$$

new provable time/space trade-off

$\rightsquigarrow$ first provable time/space trade-off for SVP

Digression: Quotient and Cosets

Let q be a positive integer. Lattices $q\mathcal{L} \subseteq \mathcal{L}$, **quotient**:

$$\mathcal{L}/q\mathcal{L} = \{c + q\mathcal{L} : c \in \mathcal{L}\} \cong \mathbb{Z}_q^n.$$

Each $c + q\mathcal{L}$ is a **coset**, there are q^n many.

- $x, y \in \mathcal{L}$ are in the same coset of $\mathcal{L}/q\mathcal{L}$ iff $x - y \in q\mathcal{L}$.

Digression: Quotient and Cosets

Let q be a positive integer. Lattices $q\mathcal{L} \subseteq \mathcal{L}$, **quotient**:

$$\mathcal{L}/q\mathcal{L} = \{c + q\mathcal{L} : c \in \mathcal{L}\} \cong \mathbb{Z}_q^n.$$

Each $c + q\mathcal{L}$ is a **coset**, there are q^n many.

- $x, y \in \mathcal{L}$ are in the same coset of $\mathcal{L}/q\mathcal{L}$ iff $x - y \in q\mathcal{L}$.

Above smoothing parameter: if $s \geqslant \eta_\varepsilon(q\mathcal{L})$ then for all $c \in \mathcal{L}$,

$$\frac{1-\varepsilon}{1+\varepsilon} \leqslant \frac{\rho_s(c + q\mathcal{L})}{\rho_s(q\mathcal{L})} \leqslant 1$$

Theorem (Micciancio and Peikert)

If $X_1, \ldots, X_k$ i.i.d from DGS with parameter $s \geqslant \sqrt{2}\eta_\varepsilon(\mathcal{L})$ s.t $\sum_i X_i \in q\mathcal{L}$ then $(X_1 + \ldots + X_k)/q$ very close to DGS with parameter $s\sqrt{k}/q$.

Sieving with two elements

Input: a list L of samples in $D_{\mathcal{L},s}$ with $s \geqslant \sqrt{2}\eta_\varepsilon(\mathcal{L})$

$L' \leftarrow$ empty list
while $|L| \geqslant 2^n + 1$ **do**
    $x \leftarrow$ a random element from L
    $y \leftarrow$ a random element from L
    if $x + y \in 2\,\mathcal{L}$ **then**
        add $(x + y)/2$ to L'
        remove x and y from L
return L'

Sieving with two elements

Input: a list L of samples in $D_{\mathcal{L},s}$ with $s \geqslant \sqrt{2}\eta_\varepsilon(\mathcal{L})$

```
L' ← empty list
while |L| ⩾ 2^n + 1 do
    x ← a random element from L
    y ← a random element from L
    if x + y ∈ 2 𝓛 then
        add (x + y)/2 to L'
        remove x and y from L
return L'
```

Analysis (above smoothing):

- sum of Gaussians $\approx$ Gaussian (Micciancio and Peikert)
  $\Rightarrow$ output L' contains independent samples from $D_{\mathcal{L},s/\sqrt{2}}$
- only cosets in $\mathcal{L}/2\mathcal{L}$ matter
- if $x \sim D_{\mathcal{L},s}$ then the coset of x mod $2\mathcal{L}$ is almost uniform

Sieving with two elements

Input: a list L of samples in $U(\mathcal{L}/2\mathcal{L})$

while $|L| \geqslant 2^n + 1$ **do**
  $x \leftarrow$ a random element from L
  $y \leftarrow$ a random element from L
  if $x + y = 2\mathcal{L}$ **then**
    remove x and y from L

pigeonhole: $|\mathcal{L}/2\mathcal{L}| = 2^n$
$\rightsquigarrow \exists x, y$ that are equal

uniformity: for every x, proba 2^{-n} that y works

Analysis (above smoothing):

- sum of Gaussians $\approx$ Gaussian (Micciancio and Peikert)
  $\Rightarrow$ output L' contains independent samples from $D_{\mathcal{L},s/\sqrt{2}}$
- only cosets in $\mathcal{L}/2\mathcal{L}$ matter
- if $x \sim D_{\mathcal{L},s}$ then the coset of x mod $2\mathcal{L}$ is almost uniform

Sieving with two elements

Input: a list L of samples in $U(\mathcal{L}/2\mathcal{L})$

while $|L| \geqslant 2^n + 1$ **do**
  $x \leftarrow$ a random element from L
  $y \leftarrow$ a random element from L
  if $x + y = 2\mathcal{L}$ **then**
    remove x and y from L

pigeonhole: $|\mathcal{L}/2\mathcal{L}| = 2^n$
$\rightsquigarrow \exists x, y$ that are equal

uniformity: for every x, proba 2^{-n} that y works

Analysis (above smoothing):

- sum of Gaussians $\approx$ Gaussian (Micciancio and Peikert)
  $\Rightarrow$ output L' contains independent samples from $D_{\mathcal{L},s/\sqrt{2}}$
- only cosets in $\mathcal{L}/2\mathcal{L}$ matter
- if $x \sim D_{\mathcal{L},s}$ then the coset of x mod $2\mathcal{L}$ is almost uniform

Problem: after removing x, y, L is not uniform anymore
$\rightsquigarrow$ show that it remains sufficiently close to uniform

Sieving with two elements

Input: a list L of samples in $U(\mathcal{L}/2\mathcal{L})$

while $|L| \geqslant 2^n + 1$ **do**
- $x \leftarrow$ a random element from L
- $y \leftarrow$ a random element from L
- **if** $x + y = 2\mathcal{L}$ **then**
  - remove x and y from L

pigeonhole: $|\mathcal{L}/2\mathcal{L}| = 2^n$
$\rightsquigarrow \exists x, y$ that are equal

uniformity: for every x, proba 2^{-n} that y works

Problem: after removing x, y, L is not uniform anymore
$\rightsquigarrow$ show that it remains sufficiently close to uniform

Take $|L| = 2^n + 2M$, produce M vectors

- space: $2^n + 2M$ (store the list)
- time: $2^n M$ ($\approx 2^n$ tries per vector)

For $M = 2^n$, space 2^n and time 2^{2n}

Sieving with k elements

Input: a list L of samples in $D_{\mathcal{L},s}$ with $s \geqslant \sqrt{2}\eta_\varepsilon(\mathcal{L})$

```
L' ← empty list
while |L| ⩾ ??? do
    x_1, ..., x_k ← random elements from L
    if x_1 + ··· + x_k ∈ q𝓛 then
        add (x_1 + ··· + x_k)/q to L'
        remove the x_i from L
return L'
```

Sieving with k elements

Input: a list L of samples in $U(\mathcal{L}/q\mathcal{L})$

while $|L| \geqslant ???$ **do**
  $x_1, \ldots, x_k \leftarrow$ random elements from L
  if $x_1 + \cdots + x_k = q\mathcal{L}$ **then**
    remove the x_i from L

Sieving with k elements

Input: a list L of samples in $U(\mathcal{L}/q\,\mathcal{L})$

while $|L| \geqslant q^{n/k}$ **do**
  $x_1, \ldots, x_k \leftarrow$ random elements from L
  if $x_1 + \cdots + x_k = q\,\mathcal{L}$ **then**
    remove the x_i from L

Cosets: $|\mathcal{L}/q\,\mathcal{L}| = q^n$
Probability analysis: $x_1 + \cdots + x_k \sim U(\mathcal{L}/q\,\mathcal{L})$

- probability q^{-n} to be $q\,\mathcal{L}$
- $\binom{|L|}{k}$ needs to be $\geqslant q^n \rightsquigarrow |L| \geqslant q^{n/k}$

Constraint: $k < q^2$ to reduce Gaussian width by a constant factor

Sieving with k elements

Input: a list L of samples in $U(\mathcal{L}/q\mathcal{L})$

while $|L| \geqslant q^{n/k}$ **do**
  $x_1, \ldots, x_k \leftarrow$ random elements from L
  if $x_1 + \cdots + x_k = q\mathcal{L}$ **then**
    remove the x_i from L

Cosets: $|\mathcal{L}/q\mathcal{L}| = q^n$
Probability analysis: $x_1 + \cdots + x_k \sim U(\mathcal{L}/q\mathcal{L})$

- probability q^{-n} to be $q\mathcal{L}$
- $\binom{|L|}{k}$ needs to be $\geqslant q^n \rightsquigarrow |L| \geqslant q^{n/k}$

Constraint: $k < q^2$ to reduce Gaussian width by a constant factor

Take $|L| = q^{n/k} + kN$, produce N vectors

- space: $q^{n/k} + kN$ (store the list)
- time: $q^n N$ ($\approx q^n$ tries per vector)

For $k = q^2 - 1$, $N = q^{n/k}$, space q^{n/q^2} and time $q^{n+n/q^2} \leqslant q^{2n}$

Time-Space Tradeoff for DGS

The real algorithm is more complicated:

- split input list into two
- probabilistic + deterministic argument to prove correctness
- exponents are much worse

Theorem (Simplified)

For $q \in [4, \sqrt{n}]$, there is an algorithm that produces q^{16n/q^2} vectors from $D_{\mathcal{L},s}$ with $s \geqslant \eta_\varepsilon(\mathcal{L})$ in time q^{13n} and space q^{16n/q^2}.

Time-Space Tradeoff for SVP

$$\text{SVP} \xrightarrow{\text{[CCL18]}} \tfrac{0.1}{q}\text{-BDD} \xrightarrow{\text{[DSR14]}} \text{DGS}$$

new provable time/space trade-off

Smooth time-space tradeoff for BDD: create a $\frac{0.1}{q}$-BDD oracle in time q^{13n}, space q^{16n/q^2}, each call takes time q^{16n/q^2}.

Gives a smooth time-space tradeoff for SVP:

Theorem

Let $n \in \mathbb{N}, q \in [4, \sqrt{n}]$ be a positive integer. Let $\mathcal{L}$ be a lattice of rank n. There is a randomized algorithm that solves SVP in time $q^{13n+o(n)}$ and in space $poly(n) \cdot q^{\frac{16n}{q^2}}$.

Time-Space Tradeoff for SVP

$$\boxed{\text{SVP}} \xrightarrow[\text{[CCL18]}]{} \boxed{\tfrac{0.1}{q}\text{-BDD}} \xrightarrow[\text{[DSR14]}]{} \boxed{\text{DGS}}$$

new provable time/space trade-off

Smooth time-space tradeoff for BDD: create a $\frac{0.1}{q}$-BDD oracle in time q^{13n}, space q^{16n/q^2}, each call takes time q^{16n/q^2}.

Gives a smooth time-space tradeoff for SVP:

Theorem

Let $n \in \mathbb{N}, q \in [4, \sqrt{n}]$ be a positive integer. Let $\mathcal{L}$ be a lattice of rank n. There is a randomized algorithm that solves SVP in time $q^{13n+o(n)}$ and in space $\text{poly}(n) \cdot q^{\frac{16n}{q^2}}$.

- $q = \sqrt{n}$: time $n^{O(n)}$ and space $\text{poly}(n)$, not as good as [Kan86].
- $q = 4$: time $2^{O(n)}$ and space $2^{O(n)}$, not as good as [ADRS15].

SVP to BDD reduction [CCL18]

Lemma (CCL18, simplified)

Given a α-BDD oracle and p an integer, one can enumerate all lattice points in a ball of radius $p\alpha\lambda_1$ using p^n queries to the oracle.

SVP to BDD reduction [CCL18]

Lemma (CCL18, simplified)

Given a α-BDD oracle and p an integer, one can enumerate all lattice points in a ball of radius $p\alpha\lambda_1$ using p^n queries to the oracle.

Solve SVP by using a α-BDD oracle:

- Set $p = \lceil \frac{1}{\alpha} \rceil$.
- Enumerate all points in a ball of radius $> \lambda_1$.

SVP to BDD reduction [CCL18]

Lemma (CCL18, simplified)

Given a α-BDD oracle and p an integer, one can enumerate all lattice points in a ball of radius $p\alpha\lambda_1$ using p^n queries to the oracle.

Solve SVP by using a α-BDD oracle:

- Set $p = \lceil \frac{1}{\alpha} \rceil$.
- Enumerate all points in a ball of radius $> \lambda_1$.

The reduction is space efficient

But $\alpha < \frac{1}{2} \implies p \geq 3 \implies$ at least 3^n queries

Quantum SVP

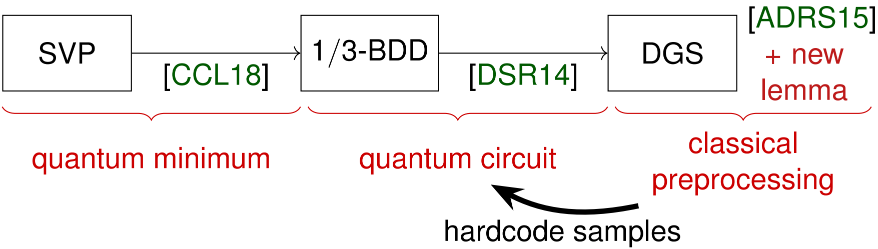
Classical SVP to BDD: do 3^n queries to $1/3$-BDD and keep minimum

$$\begin{tikzcd} \boxed{\text{SVP}} \arrow[r, "\text{[CCL18]}"'] & \boxed{1/3\text{-BDD}} \arrow[r, "\text{[DSR14]}"'] & \boxed{\text{DGS}} \end{tikzcd}$$

[ADRS15]
+ new
lemma

Quantum SVP

Classical SVP to BDD: do 3^n queries to $1/3$-BDD and keep minimum

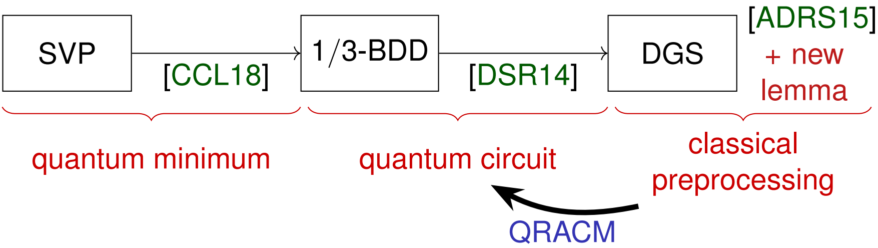


Theorem

There is a quantum algorithm that solves SVP in time $2^{0.9529n+o(n)}$, classical space $2^{0.5n+o(n)}$ and $\mathrm{poly}(n)$ qubits.

Quantum SVP

Classical SVP to BDD: do 3^n queries to $1/3$-BDD and keep minimum



Theorem

There is a quantum algorithm that solves SVP in time $2^{0.9529n+o(n)}$, classical space $2^{0.5n+o(n)}$ and $\mathrm{poly}(n)$ qubits.

Future work: use QRACM to speed-up the query time of the $1/3$-BDD.
$\rightsquigarrow$ time $2^{0.869n+o(n)}$?

DGS sampling: new lemma

- [ADRS15]: DGS of parameter $s \geqslant \sqrt{2}\eta_{1/2}(\mathcal{L})$ in time $2^{n/2}$
- BDD to DGS reduction requires $s = \eta_\varepsilon(\mathcal{L})$ for some $\varepsilon > 0$

Previous work [CCL18]: find ε such that $\eta_\varepsilon(\mathcal{L}) \geqslant \sqrt{2}\eta_{1/2}$
$\rightsquigarrow$ very small ε, larger than necessary BDD radius, too expensive BDD

DGS sampling: new lemma

- [ADRS15]: DGS of parameter $s \geqslant \sqrt{2}\eta_{1/2}(\mathcal{L})$ in time $2^{n/2}$
- BDD to DGS reduction requires $s = \eta_\varepsilon(\mathcal{L})$ for some $\varepsilon > 0$

Previous work [CCL18]: find ε such that $\eta_\varepsilon(\mathcal{L}) \geqslant \sqrt{2}\eta_{1/2}$
$\rightsquigarrow$ very small ε, larger than necessary BDD radius, too expensive BDD

New idea:

- find a well-chosen lattice $\mathcal{L} \subset \mathcal{L}' \subset \mathcal{L}/2$ such that $\eta_{\varepsilon'}(\mathcal{L}') \leqslant \eta_\varepsilon(\mathcal{L})/\sqrt{2}$ for $\varepsilon' \approx \varepsilon$ [ADRS15]
- run DGS on $\mathcal{L}'$ at $s = \eta_{1/3}(\mathcal{L}) \geqslant \sqrt{2}\eta_{1/2}(\mathcal{L}')$ [ADRS15]
- only keep samples in $\mathcal{L}$ (rejection)

DGS sampling: new lemma

- [ADRS15]: DGS of parameter $s \geqslant \sqrt{2}\eta_{1/2}(\mathcal{L})$ in time $2^{n/2}$
- BDD to DGS reduction requires $s = \eta_\varepsilon(\mathcal{L})$ for some $\varepsilon > 0$

Previous work [CCL18]: find ε such that $\eta_\varepsilon(\mathcal{L}) \geqslant \sqrt{2}\eta_{1/2}$
$\rightsquigarrow$ very small ε, larger than necessary BDD radius, too expensive BDD

New idea:

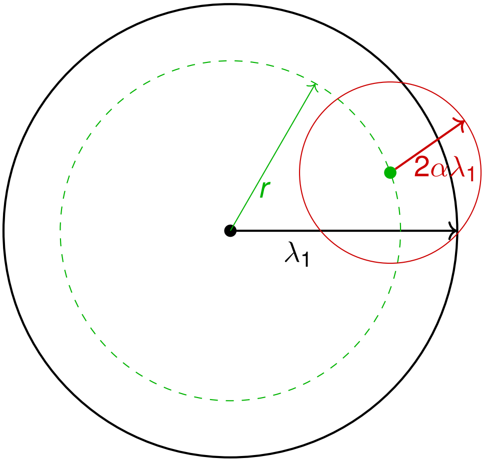
- find a well-chosen lattice $\mathcal{L} \subset \mathcal{L}' \subset \mathcal{L}/2$ such that $\eta_{\varepsilon'}(\mathcal{L}') \leqslant \eta_\varepsilon(\mathcal{L})/\sqrt{2}$ for $\varepsilon' \approx \varepsilon$ [ADRS15]
- run DGS on $\mathcal{L}'$ at $s = \eta_{1/3}(\mathcal{L}) \geqslant \sqrt{2}\eta_{1/2}(\mathcal{L}')$ [ADRS15]
- only keep samples in $\mathcal{L}$ (rejection)

Some details:

- $\mathcal{L}'$ is chosen randomly, works with high probability
- need that $|\mathcal{L}' / \mathcal{L}| \approx 2^{n/2}$ for $\varepsilon \approx \varepsilon'$
- rejection: $|\mathcal{L}' / \mathcal{L}| \approx 2^{n/2}$ slowdown, still better than previous work!
- allows to choose $\alpha = 1/3$ for BDD, improved from 0.391 [CCL18]

Faster SVP to BDD reduction

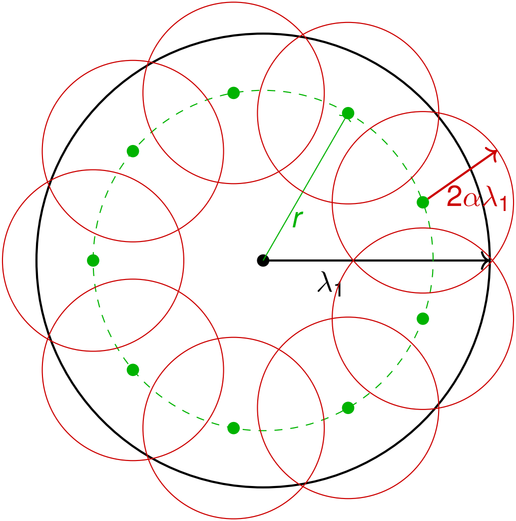
Cover the sphere of radius $\lambda_1(\mathcal{L})$ by balls of radius $2\alpha\lambda_1(\mathcal{L})$:



Use 2^n $\alpha-$BDD queries to enumerate points in balls of radius $2\alpha\lambda_1$

Faster SVP to BDD reduction

Cover the sphere of radius $\lambda_1(\mathcal{L})$ by balls of radius $2\alpha\lambda_1(\mathcal{L})$:

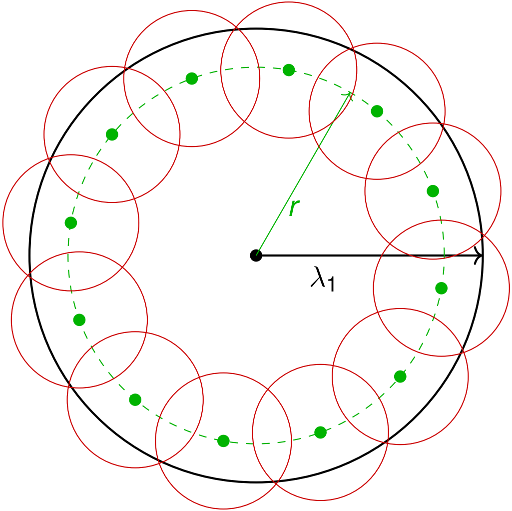


Use 2^n α−BDD queries to enumerate points in balls of radius $2\alpha\lambda_1$

Each ball covers a spherical cap.

Faster SVP to BDD reduction

Cover the sphere of radius $\lambda_1(\mathcal{L})$ by balls of radius $2\alpha\lambda_1(\mathcal{L})$:



Use 2^n $\alpha-$BDD queries to enumerate points in balls of radius $2\alpha\lambda_1$

Each ball covers a spherical cap.

Smaller α:

- More balls
- Less expensive BDD

$\rightsquigarrow$ Trade-off

Improved classical SVP

Improved SVP to BDD: do 2^n queries to 0.4103-BDD

$$\begin{tikzcd} \boxed{\text{SVP}} \arrow[r, "\text{new reduction}"'] & \boxed{0.4103\text{-BDD}} \arrow[r, "\text{[DSR14]}"'] & \boxed{\text{DGS}} & \text{[ADRS15] + new lemma} \end{tikzcd}$$

Theorem

There is a classical algorithm that solves SVP in time $2^{1.7397n+o(n)}$, classical space $2^{0.5n+o(n)}$.

Improved classical SVP

Improved SVP to BDD: do 2^n queries to 0.4103-BDD

$$\underbrace{\boxed{\text{SVP}} \xrightarrow[\text{new reduction}]{}}_{\text{quantum minimum}} \underbrace{\boxed{\text{0.4103-BDD}} \xrightarrow[\text{[DSR14]}]{}}_{\text{quantum circuit}} \underbrace{\boxed{\text{DGS}} \ \begin{array}{c}\text{[ADRS15]}\\ \text{+ new}\\ \text{lemma}\end{array}}_{\text{classical}}$$

Theorem

There is a classical algorithm that solves SVP in time $2^{1.7397n+o(n)}$, classical space $2^{0.5n+o(n)}$.

Theorem

*There is a **quantum** algorithm that solves SVP in time $2^{1.051n+o(n)}$, classical space $2^{0.5n+o(n)}$ and $\mathrm{poly}(n)$ qubits.*

Not as good as our previous $2^{0.9529n+o(n)}$ algorithm but the story does not stop here...

SVP and Generalized Kissing Number

Number of lattice points in a ball of radius r is $\leqslant c^{n+o(n)} r^n$

$$\beta(\mathcal{L}) = \text{smallest } c \text{ that works for all } r$$

- Upper bound: $\beta(\mathcal{L}) \leqslant 2^{0.401}$ [KL78]
- Conjectured to be $\beta(\mathcal{L}) \approx 1$ for most lattices

SVP and Generalized Kissing Number

Number of lattice points in a ball of radius r is $\leqslant c^{n+o(n)} r^n$

$$\beta(\mathcal{L}) = \text{smallest } c \text{ that works for all } r$$

- Upper bound: $\beta(\mathcal{L}) \leqslant 2^{0.401}$ [KL78]
- Conjectured to be $\beta(\mathcal{L}) \approx 1$ for most lattices

$$\boxed{\alpha\text{−BDD}} \xrightarrow[\text{reduce}]{\hspace{4em}} \boxed{\text{DGS}_{\eta_\varepsilon}}$$

Best known relations between α and ε depends on $\beta(\mathcal{L})$:

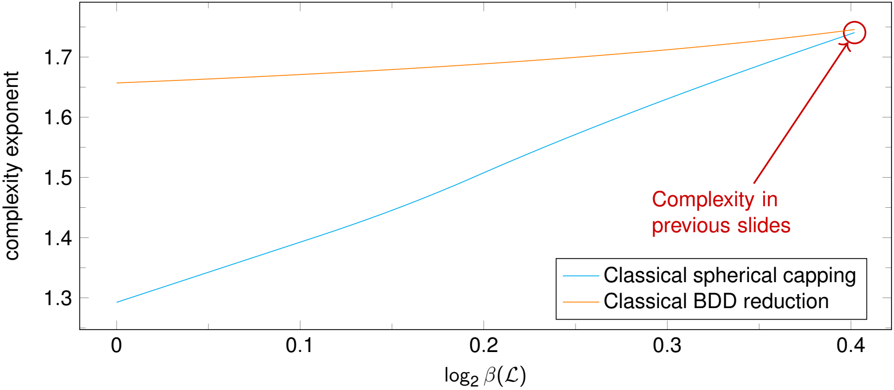
small $\beta(\mathcal{L})$ $\quad\rightsquigarrow\quad$ bigger α for fixed ε $\quad\rightsquigarrow\quad$ less expensive BDD

SVP and Generalized Kissing Number

Number of lattice points in a ball of radius r is $\leqslant c^{n+o(n)} r^n$

$$\beta(\mathcal{L}) = \text{smallest } c \text{ that works for all } r$$

- Upper bound: $\beta(\mathcal{L}) \leqslant 2^{0.401}$ [KL78]
- Conjectured to be $\beta(\mathcal{L}) \approx 1$ for most lattices

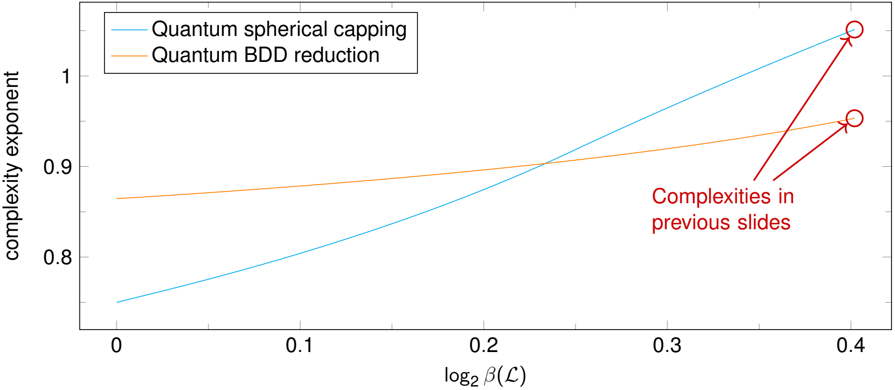


SVP and Generalized Kissing Number

Number of lattice points in a ball of radius r is $\leqslant c^{n+o(n)} r^n$

$$\beta(\mathcal{L}) = \text{smallest } c \text{ that works for all } r$$

- Upper bound: $\beta(\mathcal{L}) \leqslant 2^{0.401}$ [KL78]
- Conjectured to be $\beta(\mathcal{L}) \approx 1$ for most lattices



Details on BDD to DGS reduction

The reduction from α-BDD to DGS requires a parameter $s = \eta_\varepsilon$ for some ε that depends on α [DSR14]

Problem: we cannot compute or even approximate η_ε (this is a probably a hard problem) so how do we choose s ?

Details on BDD to DGS reduction

The reduction from α-BDD to DGS requires a parameter $s = \eta_\varepsilon$ for some ε that depends on α [DSR14]

Problem: we cannot compute or even approximate η_ε (this is a probably a hard problem) so how do we choose s ?

Idea 1: run the DGS sampler for $s_i = \delta^{-i} s_0$ for $\delta > 1$

$$\exists i \text{ st} \quad s_i \geqslant \eta_\varepsilon \geqslant \delta^{-1} s_i$$

Probably sufficient: [DSR14] most likely works with such s_i but one would need to redo the (complicated) proof...

Details on BDD to DGS reduction

The reduction from α-BDD to DGS requires a parameter $s = \eta_\varepsilon$ for some ε that depends on α [DSR14]

Problem: we cannot compute or even approximate η_ε (this is a probably a hard problem) so how do we choose s ?

Idea 1: run the DGS sampler for $s_i = \delta^{-i} s_0$ for $\delta > 1$

$$\exists i \text{ st} \quad s_i \geqslant \eta_\varepsilon \geqslant \delta^{-1} s_i$$

Probably sufficient: [DSR14] most likely works with such s_i but one would need to redo the (complicated) proof...

Idea 2: show that $s_i = \eta_{\varepsilon'}$ for some $\varepsilon^{\delta^2} \leqslant \varepsilon' \leqslant \varepsilon$
$\rightsquigarrow$ BDD radius α' is almost unchanged for $\delta = 1 + 1/n^{O(1)}$

Proof uses a new tail-bound that involves $\beta(\mathcal{L})$ and a new lower bound on $\eta_{\varepsilon^{\delta^2}}$

Conclusions and Future work

Provable SVP:

- classical: time $2^{1.7397n+o(n)}$, space $2^{0.5n+o(n)}$
- quantum: $2^{0.9529n+o(n)}$, space $2^{0.5n+o(n)}$ and $\mathrm{poly}(n)$ qubits
- first time/space tradeoff: time q^{13n}, space q^{16n/q^2} for $q \in [4, \sqrt{n}]$
- studied dependency on $\beta(\mathcal{L})$, generalized kissing number

Conclusions and Future work

Provable SVP:

- classical: time $2^{1.7397n+o(n)}$, space $2^{0.5n+o(n)}$
- quantum: $2^{0.9529n+o(n)}$, space $2^{0.5n+o(n)}$ and $\mathrm{poly}(n)$ qubits
- first time/space tradeoff: time q^{13n}, space q^{16n/q^2} for $q \in [4, \sqrt{n}]$
- studied dependency on $\beta(\mathcal{L})$, generalized kissing number

Open problems:

- Show that random lattices satisfy $\beta(\mathcal{L}) \approx 1$?
- Fill the gap between provable and heuristic algorithms for sieving?
- Exploit the subexponential space regime in our trade-off for SVP?
- $2^{O(n)}$ time, $2^{o(n)}$ space algorithm for DGS at smoothing parameter?