Lattice-based crypto, part 1:
Algorithmic problems over lattices

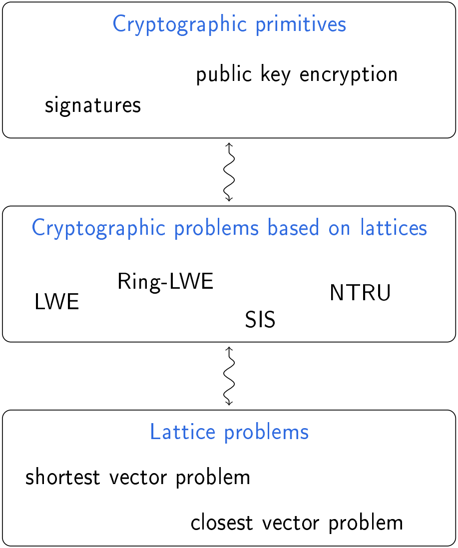
Alice Pellet--Mary

CNRS and university of Bordeaux, France

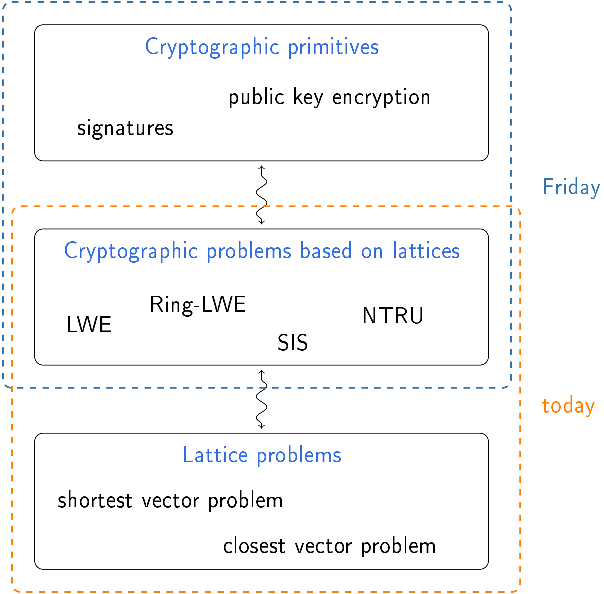
Summer school in post-quantum cryptography 2022

1-5 August 2022, Budapest

Plan of the talks



Plan of the talks

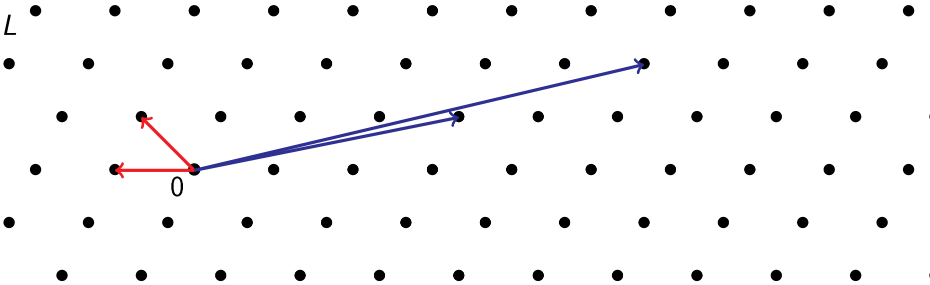


Outline of the talk

1. Lattices and lattice problems
2. Cryptographic problems based on lattices
3. Algorithms

Outline of the talk

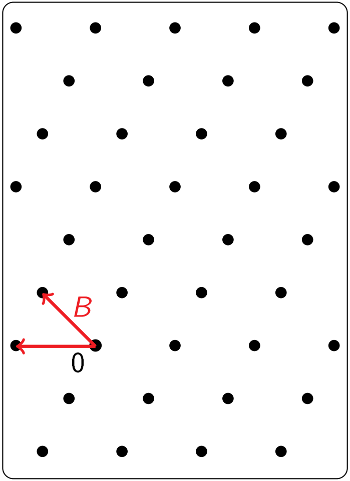
Lattices



- $L = \mathcal{L}(B) = \{Bx \mid x \in \mathbb{Z}^n\}$ is a lattice
- $B \in \mathrm{GL}_n(\mathbb{R})$ is a basis
- n is the dimension of L (or rank)

Representing a lattice

Representation of a lattice L:
a basis $B \in \mathbb{Z}^{n \times n}$ of L



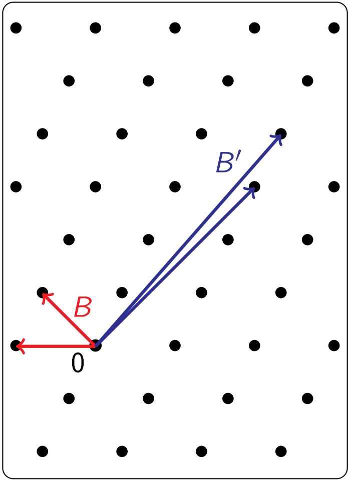
Representing a lattice

Representation of a lattice L:

a basis $B \in \mathbb{Z}^{n \times n}$ of L

Difficulty:

- the basis B is not unique
- some choices of B can make some algorithmic problems easier



Representing a lattice

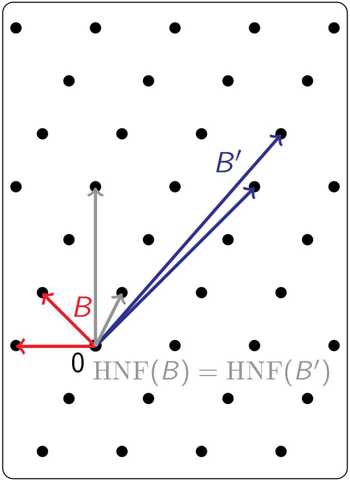
Representation of a lattice L:
a basis $B \in \mathbb{Z}^{n\times n}$ of L

Difficulty:

- the basis B is not unique
- some choices of B can make some algorithmic problems easier

Solution: take the Hermite Normal Form (HNF) of any B

- it is unique $(\mathrm{HNF}(B) = \mathrm{HNF}(B'))$
- it is efficiently computable



Representing a lattice

Representation of a lattice L:
a basis $B \in \mathbb{Z}^{n \times n}$ of L

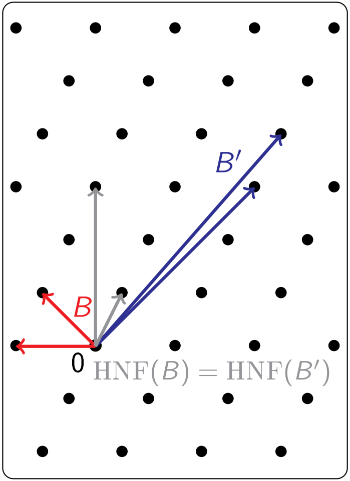
Difficulty:

- the basis B is not unique
- some choices of B can make some algorithmic problems easier

Solution: take the Hermite Normal Form (HNF) of any B

- it is unique $(\mathrm{HNF}(B) = \mathrm{HNF}(B'))$
- it is efficiently computable

$\Rightarrow$ canonical representation of L
(i.e., worse basis ever)



Algorithmic problems on lattices

Input: the HNF basis of any lattice

Example of problems:

(1) Testing equality of lattices
(2) Testing inclusion of lattices
(3) Intersecting two lattices
(4) Computing a short vector of a lattice
(5) Computing a lattice vector close to a target

Algorithmic problems on lattices

Input: the HNF basis of any lattice

Example of problems:

(1) Testing equality of lattices
(2) Testing inclusion of lattices
(3) Intersecting two lattices
(4) Computing a short vector of a lattice
(5) Computing a lattice vector close to a target

Quiz: which ones are easy or hard?

easy: polynomial time — hard: no polynomial time algorithm known

Algorithmic problems on lattices

Input: the HNF basis of any lattice

Example of problems:

(1) Testing equality of lattices ⇒ easy
(2) Testing inclusion of lattices ⇒ easy
(3) Intersecting two lattices ⇒ easy
(4) Computing a short vector of a lattice ⇒ hard
(5) Computing a lattice vector close to a target ⇒ hard

Quiz: which ones are easy or hard?

easy: polynomial time　　hard: no polynomial time algorithm known

Testing inclusion / equality

Exercise

Given $B_1, B_2 \in \mathrm{GL}_n(\mathbb{R})$, how do you test if

1. $\mathcal{L}(B_1) \subseteq \mathcal{L}(B_2)$
2. $\mathcal{L}(B_1) = \mathcal{L}(B_2)$

Testing inclusion / equality

Exercise

Given $B_1, B_2 \in \mathrm{GL}_n(\mathbb{R})$, how do you test if

1. $\mathcal{L}(B_1) \subseteq \mathcal{L}(B_2)$
2. $\mathcal{L}(B_1) = \mathcal{L}(B_2)$

Solution:

1.

$$\mathcal{L}(B_1) \subseteq \mathcal{L}(B_2) \Leftrightarrow B_1 = B_2 \cdot X \quad \text{for some } X \in \mathbb{Z}^{n \times n}$$
$$\Leftrightarrow B_1 \cdot B_2^{-1} \in \mathbb{Z}^{n \times n}$$

Testing inclusion / equality

Exercise

Given $B_1, B_2 \in \mathrm{GL}_n(\mathbb{R})$, how do you test if

1. $\mathcal{L}(B_1) \subseteq \mathcal{L}(B_2)$
2. $\mathcal{L}(B_1) = \mathcal{L}(B_2)$

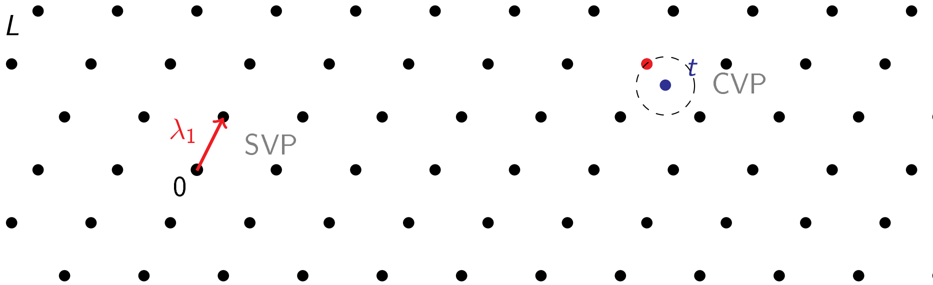
Solution:

1.

$$\begin{aligned} \mathcal{L}(B_1) \subseteq \mathcal{L}(B_2) &\Leftrightarrow B_1 = B_2 \cdot X \quad \text{for some } X \in \mathbb{Z}^{n \times n} \\ &\Leftrightarrow B_1 \cdot B_2^{-1} \in \mathbb{Z}^{n \times n} \end{aligned}$$

2. $\mathcal{L}(B_1) = \mathcal{L}(B_2) \Leftrightarrow \mathcal{L}(B_1) \subseteq \mathcal{L}(B_2)$ and $\mathcal{L}(B_2) \subseteq \mathcal{L}(B_1)$

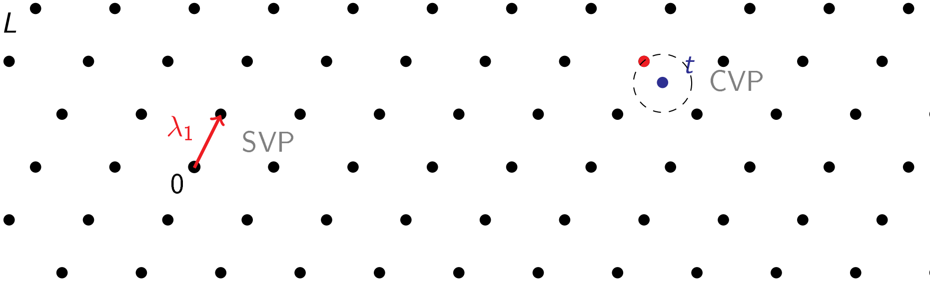
(Hard) lattice problems



SVP : Shortest Vector Problem
(input: HNF basis of L)

CVP : Closest Vector Problem
(input: HNF basis of L and target t)

(Hard) lattice problems



SVP : Shortest Vector Problem
(input: HNF basis of L)

CVP : Closest Vector Problem
(input: HNF basis of L and target t)

Supposedly hard to solve when n is large
(even with a quantum computer)

How hard is SVP/CVP?

In theory: best algorithm has asymptotic complexity $2^{c \cdot n + o(n)}$
(for some $c \approx 0.292$, or $c \approx 0.265$ for quantum computers [Laa15])

$\Rightarrow$ not polynomial

[Laa15] Laarhoven. Search problems in cryptography.

How hard is SVP/CVP?

In theory: best algorithm has asymptotic complexity $2^{c \cdot n + o(n)}$
(for some $c \approx 0.292$, or $c \approx 0.265$ for quantum computers [Laa15])

$$\Rightarrow \text{not polynomial}$$

In practice:

- $n = 2 \rightsquigarrow$ easy, very efficient in practice

[Laa15] Laarhoven. Search problems in cryptography.

How hard is SVP/CVP?

In theory: best algorithm has asymptotic complexity $2^{c \cdot n + o(n)}$
(for some $c \approx 0.292$, or $c \approx 0.265$ for quantum computers [Laa15])

$$\Rightarrow \text{not polynomial}$$

In practice:

- $n = 2 \rightsquigarrow$ easy, very efficient in practice
- up to $n = 60$ or $n = 80 \rightsquigarrow$ a few minutes on a personal laptop

[Laa15] Laarhoven. Search problems in cryptography.

How hard is SVP/CVP?

In theory: best algorithm has asymptotic complexity $2^{c \cdot n + o(n)}$
(for some $c \approx 0.292$, or $c \approx 0.265$ for quantum computers [Laa15])

$\Rightarrow$ not polynomial

In practice:

- $n = 2 \rightsquigarrow$ easy, very efficient in practice
- up to $n = 60$ or $n = 80 \rightsquigarrow$ a few minutes on a personal laptop
- up to $n = 180 \rightsquigarrow$ few days on big computers with good code [DSW21]

[Laa15] Laarhoven. Search problems in cryptography.

[DSW21] Ducas, Stevens, van Woerden. Advanced Lattice Sieving on GPUs, with Tensor Cores.

How hard is SVP/CVP?

In theory: best algorithm has asymptotic complexity $2^{c \cdot n + o(n)}$
(for some $c \approx 0.292$, or $c \approx 0.265$ for quantum computers [Laa15])

$\Rightarrow$ not polynomial

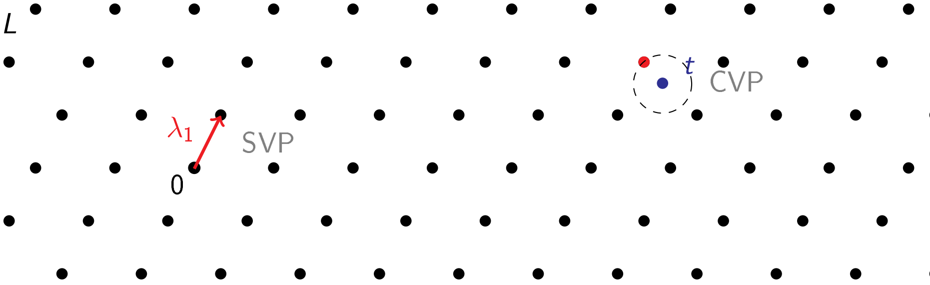
In practice:

- $n = 2 \rightsquigarrow$ easy, very efficient in practice
- up to $n = 60$ or $n = 80 \rightsquigarrow$ a few minutes on a personal laptop
- up to $n = 180 \rightsquigarrow$ few days on big computers with good code [DSW21]
- from $n = 500$ to $n = 1000 \rightsquigarrow$ cryptography

[Laa15] Laarhoven. Search problems in cryptography.

[DSW21] Ducas, Stevens, van Woerden. Advanced Lattice Sieving on GPUs, with Tensor Cores.

Approximate lattice problems



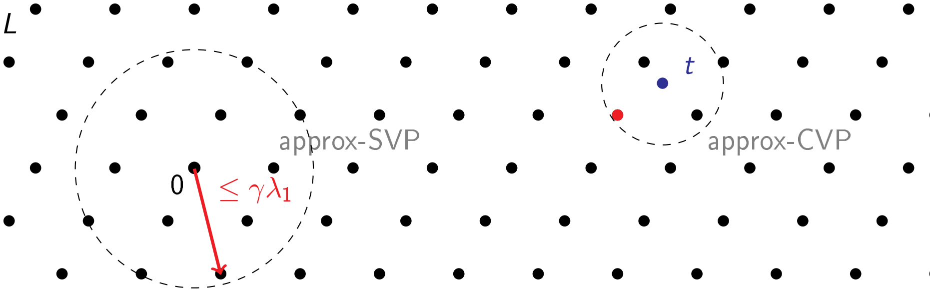
SVP : Shortest Vector Problem

CVP : Closest Vector Problem

Supposedly hard to solve when n is large

(even with a quantum computer)

Approximate lattice problems



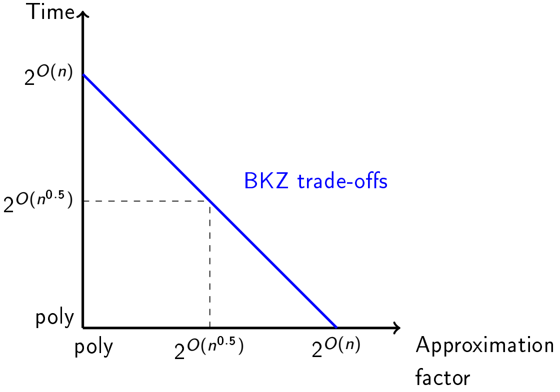
approx-SVP : Shortest Vector Problem

approx-CVP : Closest Vector Problem

Supposedly hard to solve when n is large
when the approximation factor is small ($\mathrm{poly}(n)$)
(even with a quantum computer)

Asymptotic hardness of approx-SVP/CVP

Best Time/Approximation trade-off for SVP, CVP (even quantumly):
BKZ algorithm [Sch87,SE94]

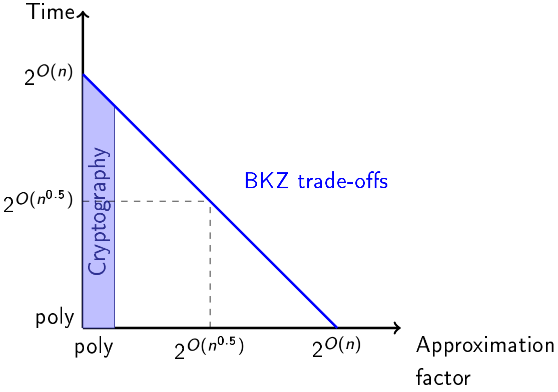


[Sch87] C.-P. Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. TCS.

[SE94] C.-P. Schnorr and M. Euchner. Lattice basis reduction: improved practical algorithms and solving subset sum problems. Mathematical programming.

Asymptotic hardness of approx-SVP/CVP

Best Time/Approximation trade-off for SVP, CVP (even quantumly):
BKZ algorithm [Sch87,SE94]



[Sch87] C.-P. Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. TCS.

[SE94] C.-P. Schnorr and M. Euchner. Lattice basis reduction: improved practical algorithms and solving subset sum problems. Mathematical programming.

Section's conclusion

γ-SVP and γ-CVP: (i.e., approx SVP/CVP with approx factor γ)

- best algorithm so far requires $2^{\Omega(n)}$ time if $\gamma = \mathrm{poly}(n)$
- even quantumly

Section's conclusion

γ-SVP and γ-CVP: (i.e., approx SVP/CVP with approx factor γ)

- best algorithm so far requires $2^{\Omega(n)}$ time if $\gamma = \mathrm{poly}(n)$
- even quantumly

Zoo of lattice problems: many variants $\Rightarrow$ how do they compare?

Section's conclusion

γ-SVP and γ-CVP: (i.e., approx SVP/CVP with approx factor γ)

- best algorithm so far requires $2^{\Omega(n)}$ time if $\gamma = \mathrm{poly}(n)$
- even quantumly

Zoo of lattice problems: many variants $\Rightarrow$ how do they compare?

- exact vs approx
- search vs decision
- one short vector vs a short basis

Section's conclusion

γ-SVP and γ-CVP: (i.e., approx SVP/CVP with approx factor γ)

- best algorithm so far requires $2^{\Omega(n)}$ time if $\gamma = \mathrm{poly}(n)$
- even quantumly

Zoo of lattice problems: many variants $\Rightarrow$ how do they compare?

- exact vs approx
- search vs decision
- one short vector vs a short basis
- see [Ste16] for a very nice picture

[Ste16] Stephens-Davidowitz. Dimension-preserving reductions between lattice problems.
http://www.noahsd.com/latticeproblems.pdf

Outline of the talk

Limitations of SVP (and CVP)

SVP and CVP are hard in the worst case

Limitations of SVP (and CVP)

SVP and CVP are hard in the worst case

- no efficient algorithm that works for any lattice

Limitations of SVP (and CVP)

SVP and CVP are hard in the worst case

- no efficient algorithm that works for any lattice
- but for some lattice it might be easier
  - demo

Limitations of SVP (and CVP)

SVP and CVP are hard in the worst case

- no efficient algorithm that works for any lattice
- but for some lattice it might be easier
  - demo

For crypto, we need problems that are hard on average

(i.e., for a random instance, the problem is hard with overwhelming probability)

SIS

The SIS problem

Notations: q, B integers, $1 \leq B \ll q$, $\mathbb{Z}_q := \mathbb{Z}/q\mathbb{Z}$

SIS (Short Integer Solution) [Ajt96]

Parameters: B and q

Problem: Given $A \leftarrow \text{Uniform}(\mathbb{Z}_q^{m \times n})$ (with $n \log q < m$)

Find $x \in \mathbb{Z}^m$ s.t. $x^T A = 0 \bmod q$ with $\|x\| \leq B$ and $x \neq 0$.

[Ajt96] Ajtai. Generating hard instances of lattice problems. STOC.

The SIS problem

Notations: q, B integers, $1 \leq B \ll q$, $\mathbb{Z}_q := \mathbb{Z}/q\mathbb{Z}$

SIS (Short Integer Solution) [Ajt96]

Parameters: B and q

Problem: Given $A \leftarrow \text{Uniform}(\mathbb{Z}_q^{m \times n})$ (with $n \log q < m$)

Find $x \in \mathbb{Z}^m$ s.t. $x^T A = 0 \bmod q$ with $\|x\| \leq B$ and $x \neq 0$.

Solving SIS with non-negligible probability	$\gtrsim$	Solving approx-SVP in any lattice of rank n

[Ajt96] Ajtai. Generating hard instances of lattice problems. STOC.

The SIS problem

Notations: q, B integers, $1 \leq B \ll q$, $\mathbb{Z}_q := \mathbb{Z}/q\mathbb{Z}$

SIS (Short Integer Solution) [Ajt96]

Parameters: B and q

Problem: Given $A \leftarrow \text{Uniform}(\mathbb{Z}_q^{m \times n})$ (with $n \log q < m$)

Find $x \in \mathbb{Z}^m$ s.t. $x^T A = 0 \bmod q$ with $\|x\| \leq B$ and $x \neq 0$.

Solving approx-SVP in any lattice lattice of rank m	$\geq$	Solving SIS with non-negligible probability	$\gtrsim$	Solving approx-SVP in any lattice of rank n

[Ajt96] Ajtai. Generating hard instances of lattice problems. STOC.

SIS is as hard as worst-case lattice problems

Theorem [Ajt96]

For any $m = \mathrm{poly}(n)$ and $B > 0$ and sufficiently large $q \geq B \cdot \mathrm{poly}(n)$, there is a reduction from solving SIS to solving γ-SIVP on arbitrary n-dimensional lattice, for some approximation factor $\gamma = B \cdot \mathrm{poly}(n)$.

(SIVP = shortest independent vectors problems.

Objective: find n short linearly independent vectors in the lattice)

[Ajt96] Ajtai. Generating hard instances of lattice problems. STOC.

SIS is as hard as worst-case lattice problems

Theorem [Ajt96]

For any $m = \mathrm{poly}(n)$ and $B > 0$ and sufficiently large $q \geq B \cdot \mathrm{poly}(n)$, there is a reduction from solving SIS to solving γ-SIVP on arbitrary n-dimensional lattice, for some approximation factor $\gamma = B \cdot \mathrm{poly}(n)$.

(SIVP = shortest independent vectors problems.
Objective: find n short linearly independent vectors in the lattice)

- the poly quantities have been improved in more recent works
- see [Pei16] for a survey

[Pei16] Peikert. A decade of lattice cryptography. Foundations and trends in theoretical computer science

SIS is a lattice problem

SIS (Short Integer Solution)

Given A $\leftarrow$ Uniform($\mathbb{Z}_q^{m\times n}$) (with $n \log q < m$)

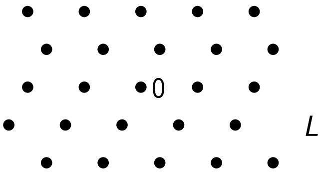
Find $x \in \mathbb{Z}^m$ with $\|x\| \leq B$ and $x \neq 0$ s.t. $x^T A = 0 \bmod q$.

SIS is a lattice problem

SIS (Short Integer Solution)

Given $A \leftarrow \text{Uniform}(\mathbb{Z}_q^{m \times n})$ (with $n \log q < m$)

Find $x \in \mathbb{Z}^m$ with $\|x\| \leq B$ and $x \neq 0$ s.t. $x^T A = 0 \bmod q$.



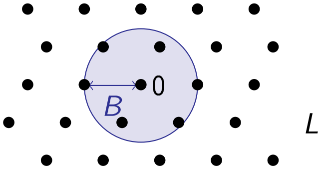
$$L = \{x \in \mathbb{Z}^m \mid x^T A = 0 \bmod q\}$$

SIS is a lattice problem

SIS (Short Integer Solution)

Given $A \leftarrow \text{Uniform}(\mathbb{Z}_q^{m\times n})$ (with $n \log q < m$)

Find $x \in \mathbb{Z}^m$ with $\|x\| \leq B$ and $x \neq 0$ s.t. $x^T A = 0 \bmod q$.



$$L = \{x \in \mathbb{Z}^m \,|\, x^T A = 0 \bmod q\}$$

SIS $\approx$ approx-SVP in L

LWE

The LWE problem

Notations: q, B integers, $1 \leq B \ll q$, $\mathbb{Z}_q := \mathbb{Z}/q\mathbb{Z}$

LWE (Learning With Errors) [Reg05]

Sample $A \leftarrow \text{Uniform}(\mathbb{Z}_q^{m\times n})$ and s, $e \leftarrow \text{Uniform}(\{-B, \cdots, B\}^n)$

Given A and b, where $b := A s + e \bmod q$

Recover s or e

[Reg05] Regev. On lattices, learning with errors, random linear codes, and cryptography. STOC.

The LWE problem

Notations: q, B integers, $1 \leq B \ll q$, $\mathbb{Z}_q := \mathbb{Z}/q\mathbb{Z}$

LWE (Learning With Errors) [Reg05]

Sample $A \leftarrow \mathrm{Uniform}(\mathbb{Z}_q^{m \times n})$ and s, $e \leftarrow \mathrm{Uniform}(\{-B, \cdots, B\}^n)$

Given A and b, where $b := A s + e \bmod q$

Recover s or e

Remark. Sometimes s is uniform in $\mathbb{Z}_q$ (not small)

- this is (almost) equivalent
- prove it (*hint:* you are allowed to change m)

[Reg05] Regev. On lattices, learning with errors, random linear codes, and cryptography. STOC.

The LWE problem

Notations: q, B integers, $1 \leq B \ll q$, $\mathbb{Z}_q := \mathbb{Z}/q\mathbb{Z}$

LWE (Learning With Errors) [Reg05]

Sample A $\leftarrow$ Uniform($\mathbb{Z}_q^{m \times n}$) and s, e $\leftarrow$ Uniform($\{-B, \cdots, B\}^n$)

Given A and b, where $b := A \, s + e \bmod q$

Recover s or e

Solving LWE with non-negligible probability $\gtrsim$ (quantumly!) Solving approx-SVP in any lattice of rank n

[Reg05] Regev. On lattices, learning with errors, random linear codes, and cryptography. STOC.

The LWE problem

Notations: q, B integers, $1 \leq B \ll q$, $\mathbb{Z}_q := \mathbb{Z}/q\mathbb{Z}$

LWE (Learning With Errors) [Reg05]

Sample $A \leftarrow \text{Uniform}(\mathbb{Z}_q^{m\times n})$ and s, $e \leftarrow \text{Uniform}(\{-B, \cdots, B\}^n)$

Given A and b, where $b := A s + e \bmod q$

Recover s or e

Solving approx-SVP in any lattice of rank m	$\gtrsim$	Solving LWE with non-negligible probability	$\gtrsim$ quantumly!	Solving approx-SVP in any lattice of rank n

[Reg05] Regev. On lattices, learning with errors, random linear codes, and cryptography. STOC.

LWE is quantumly as hard as worst-case lattice problems

Theorem [Reg05]

For any $m = \mathrm{poly}(n)$, modulus $q \leq 2^{\mathrm{poly}(n)}$ and $B \geq 2\sqrt{n}$, there is a quantum reduction from solving LWE to solving γ-SIVP on arbitrary n-dimensional lattice, for some approximation factor $\gamma = \tilde{O}(n \cdot q/B)$.

⚠ the reduction is for a variant of LWE where s and e are sampled from a discrete Gaussian distribution of parameter B ⚠

[Reg05] Regev. On lattices, learning with errors, random linear codes, and cryptography. STOC.

LWE is quantumly as hard as worst-case lattice problems

Theorem [Reg05]

For any $m = \mathrm{poly}(n)$, modulus $q \leq 2^{\mathrm{poly}(n)}$ and $B \geq 2\sqrt{n}$, there is a quantum reduction from solving LWE to solving γ-SIVP on arbitrary n-dimensional lattice, for some approximation factor $\gamma = \tilde{O}(n \cdot q/B)$.

⚠ the reduction is for a variant of LWE where s and e are sampled from a discrete Gaussian distribution of parameter B ⚠

Remark: the reduction can be made fully classical [Pei09, BLPRS13]

[Pei09] Peikert. Public-key cryptosystems from the worst-case shortest vector problem. STOC.

[BLPRS13] Brakerski, Langlois, Peikert, Regev, and Stehlé. Classical hardness of learning with errors. STOC

LWE is a lattice problem

LWE (Learning With Errors)

Sample $A \leftarrow \text{Uniform}(\mathbb{Z}_q^{n\times n})$ and s, $e \leftarrow \text{Uniform}(\{-B, \cdots, B\}^n)$

Given A and b, where $b := A\, s + e \bmod q$

Recover s or e

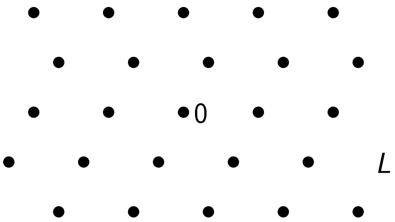
LWE is a lattice problem

LWE (Learning With Errors)

Sample $A \leftarrow \text{Uniform}(\mathbb{Z}_q^{n \times n})$ and s, $e \leftarrow \text{Uniform}(\{-B, \cdots, B\}^n)$

Given A and b, where $b := A s + e \bmod q$

Recover s or e



$$L = \{x \in \mathbb{Z}^n \mid \exists s \in \mathbb{Z}^n, As = x \bmod q\}$$

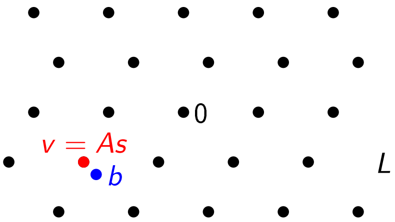
LWE is a lattice problem

LWE (Learning With Errors)

Sample $A \leftarrow \text{Uniform}(\mathbb{Z}_q^{n\times n})$ and s, $e \leftarrow \text{Uniform}(\{-B, \cdots, B\}^n)$

Given A and b, where $b := A \, s + e \mod q$

Recover s or e



$$L = \{x \in \mathbb{Z}^n \mid \exists s \in \mathbb{Z}^n, As = x \bmod q\}$$

$$b = v + e,$$

where $v \in L$ and e small

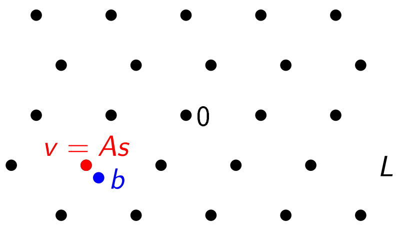
LWE is a lattice problem

LWE (Learning With Errors)

Sample $A \leftarrow \text{Uniform}(\mathbb{Z}_q^{n\times n})$ and s, $e \leftarrow \text{Uniform}(\{-B,\cdots,B\}^n)$

Given A and b, where $b := A s + e \bmod q$

Recover s or e



$$L = \{x \in \mathbb{Z}^n \,|\, \exists s \in \mathbb{Z}^n, As = x \bmod q\}$$

$$b = v + e,$$

where $v \in L$ and e small

LWE $\approx$ CVP in L

Summary on SIS and LWE

SIS and LWE are average-case problems

Summary on SIS and LWE

SIS and LWE are average-case problems
$\Rightarrow$ Good for crypto
(negligible probability to sample a weak key)

Summary on SIS and LWE

SIS and LWE are average-case problems
$\Rightarrow$ Good for crypto
(negligible probability to sample a weak key)

SIS $\overset{\sim}{\longleftrightarrow}$ average case SVP

LWE $\overset{\sim}{\longleftrightarrow}$ average case CVP

Decision variant of LWE

decision-LWE

Sample $A \leftarrow \text{Uniform}(\mathbb{Z}_q^{n\times m})$ and s, $e \leftarrow \text{Uniform}(\{-B, \cdots, B\}^n)$

Given A and b, where

$$b := A s + e \bmod q \quad \text{or} \quad b \leftarrow \text{Uniform}(\mathbb{Z}_q^n)$$

Guess whether b is uniform or not.

Decision variant of LWE

decision-LWE

Sample $A \leftarrow \text{Uniform}(\mathbb{Z}_q^{n\times m})$ and s, $e \leftarrow \text{Uniform}(\{-B, \cdots, B\}^n)$

Given A and b, where

$$b := A s + e \bmod q \quad \text{or} \quad b \leftarrow \text{Uniform}(\mathbb{Z}_q^n)$$

Guess whether b is uniform or not.

decision LWE $\overset{\sim}{\Longleftrightarrow}$ (search) LWE

Decision variant of LWE

decision-LWE

Sample $A \leftarrow \text{Uniform}(\mathbb{Z}_q^{n \times m})$ and s, $e \leftarrow \text{Uniform}(\{-B, \cdots, B\}^n)$

Given A and b, where

$$b := A s + e \bmod q \quad \text{or} \quad b \leftarrow \text{Uniform}(\mathbb{Z}_q^n)$$

Guess whether b is uniform or not.

decision LWE $\overset{\sim}{\Longleftrightarrow}$ (search) LWE

$\Rightarrow$ decision problems can be easier to use for crypto

LWE vs SIS

decision-LWE $\overset{\sim}{\Longleftrightarrow}$ (search) LWE $\underset{\text{quantum}}{\overset{\sim}{\Longleftrightarrow}}$ SIS

LWE vs SIS

$$\text{decision-LWE} \overset{\sim}{\Longleftrightarrow} \text{(search) LWE} \underset{\text{quantum}}{\overset{\sim}{\Longleftrightarrow}} \text{SIS}$$

Exercise

Prove that decision-LWE $\leq$ SIS

Hint: Assume that we know x^T small such that $x^T A = 0 \bmod q$

How to distinguish

$$b := A s + e \bmod q \quad \text{vs} \quad b \leftarrow \text{Uniform}(\mathbb{Z}_q^n)$$

LWE vs SIS

$$\text{decision-LWE} \overset{\sim}{\Longleftrightarrow} \text{(search) LWE} \underset{\text{quantum}}{\overset{\sim}{\Longleftrightarrow}} \text{SIS}$$

Exercise

Prove that decision-LWE $\leq$ SIS

Hint: Assume that we know x^T small such that $x^T A = 0 \bmod q$

How to distinguish

$$b := A s + e \bmod q \quad \text{vs} \quad b \leftarrow \text{Uniform}(\mathbb{Z}_q^n)$$

Solution: Compute $x^T b$, this gives

$$\underbrace{x^T A}_{0} s + x^T e = \underbrace{x^T e}_{\text{small}} \bmod q \quad \text{vs} \quad \underbrace{x^T b}_{\text{uniform}}$$

Section's summary

(decision) LWE / SIS:

- all somewhat equivalent (quantumly)

Section's summary

(decision) LWE / SIS:

- all somewhat equivalent (quantumly)
- as hard as worst-case lattice problems

Section's summary

(decision) LWE / SIS:

- all somewhat equivalent (quantumly)
- as hard as worst-case lattice problems
  - no major flaw in the design
  - but cryptographic constructions choose smaller parameters than the ones needed for the reductions

Section's summary

(decision) LWE / SIS:

- all somewhat equivalent (quantumly)
- as hard as worst-case lattice problems
  - no major flaw in the design
  - but cryptographic constructions choose smaller parameters than the ones needed for the reductions
- best known algorithm has time $2^{\Omega(n)}$ (for well chosen parameters q and B)

Section's summary

(decision) LWE / SIS:

- all somewhat equivalent (quantumly)
- as hard as worst-case lattice problems
  - no major flaw in the design
  - but cryptographic constructions choose smaller parameters than the ones needed for the reductions
- best known algorithm has time $2^{\Omega(n)}$ (for well chosen parameters q and B)
  - it transforms LWE and SIS into SVP/CVP instances

Section's summary

(decision) LWE / SIS:

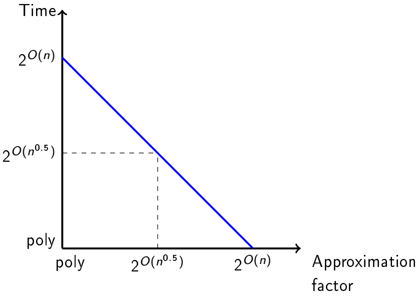
- all somewhat equivalent (quantumly)
- as hard as worst-case lattice problems
  - no major flaw in the design
  - but cryptographic constructions choose smaller parameters than the ones needed for the reductions
- best known algorithm has time $2^{\Omega(n)}$ (for well chosen parameters q and B)
  - it transforms LWE and SIS into SVP/CVP instances
- very useful survey [Pei16]

[Pei16] Peikert. A decade of lattice cryptography. Foundations and trends in theoretical computer science

Outline of the talk

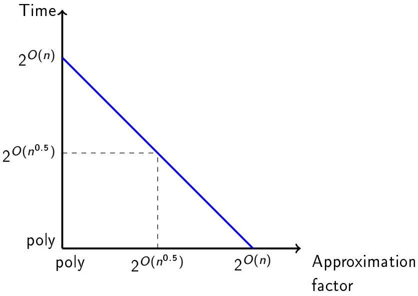
Various SVP algorithms

BKZ trade-offs



Various SVP algorithms

BKZ trade-offs

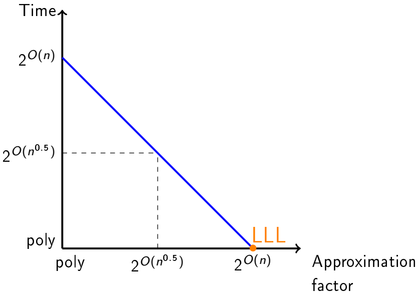


Lagrange-Gauss algorithm: dim 2

- exact SVP
- polynomial time

Various SVP algorithms

BKZ trade-offs



Lagrange-Gauss algorithm: dim 2

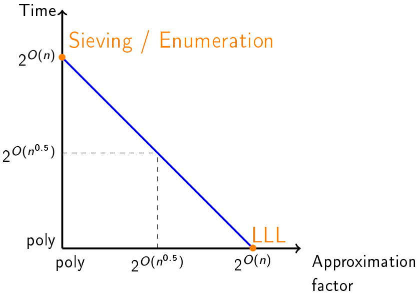
- exact SVP
- polynomial time

LLL algorithm: dim n

- γ-SVP with $\gamma = 2^n$
- polynomial time

Various SVP algorithms

BKZ trade-offs



Lagrange-Gauss algorithm: dim 2

- exact SVP
- polynomial time

LLL algorithm: dim n

- γ-SVP with $\gamma = 2^n$
- polynomial time

Sieving algorithm: dim n

- exact SVP
- time $2^{O(n)}$

Lagrange-Gauss algorithm

video

Lagrange-Gauss algorithm

video

Theorem: the algorithm

- finds a shortest vector of L
- runs in polynomial time

The LLL algorithm [LLL82]

Input: basis $B = (b_1, \ldots, b_n)$

[LLL82] Lenstra, Lenstra, and Lovász. Factoring polynomials with rational coefficients. Mathematische annalen.

The LLL algorithm [LLL82]

Input: basis $B = (b_1, \ldots, b_n)$

Main idea: improve the basis locally on blocks of dimension 2
(using Lagrange-Gauss algorithm)

[LLL82] Lenstra, Lenstra, and Lovász. Factoring polynomials with rational coefficients. Mathematische annalen.

The LLL algorithm [LLL82]

Input: basis $B = (b_1, \ldots, b_n)$

Main idea: improve the basis locally on blocks of dimension 2

(using Lagrange-Gauss algorithm)

Algorithm:

- while there exist i such that $\|b_i\|_2 > \lambda_1(L_i)$
  (L_i is roughly the lattice spanned by (b_i, b_{i+1}))
  - run Lagrange-Gauss on L_i

[LLL82] Lenstra, Lenstra, and Lovász. Factoring polynomials with rational coefficients. Mathematische annalen.

The LLL algorithm [LLL82]

Input: basis $B = (b_1, \ldots, b_n)$

Main idea: improve the basis locally on blocks of dimension 2
(using Lagrange-Gauss algorithm)

Algorithm:

- while there exist i such that $\|b_i\|_2 > \lambda_1(L_i)$
  (L_i is roughly the lattice spanned by (b_i, b_{i+1}))
  - run Lagrange-Gauss on L_i

This algorithm

- finds $v \in L$ with $\|v\|_2 \leq 2^n \cdot \lambda_1(L)$

[LLL82] Lenstra, Lenstra, and Lovász. Factoring polynomials with rational coefficients. Mathematische annalen.

The LLL algorithm [LLL82]

Input: basis $B = (b_1, \ldots, b_n)$

Main idea: improve the basis locally on blocks of dimension 2
(using Lagrange-Gauss algorithm)

Algorithm:

- while there exist i such that $\|b_i\|_2 > \lambda_1(L_i)$
  (L_i is roughly the lattice spanned by (b_i, b_{i+1}))
  - run Lagrange-Gauss on L_i

This algorithm

- finds $v \in L$ with $\|v\|_2 \leq 2^n \cdot \lambda_1(L)$
- does not run in polynomial time

[LLL82] Lenstra, Lenstra, and Lovász. Factoring polynomials with rational coefficients. Mathematische annalen.

The LLL algorithm [LLL82]

Input: basis $B = (b_1, \ldots, b_n)$

Main idea: improve the basis locally on blocks of dimension 2
(using Lagrange-Gauss algorithm)

Algorithm:

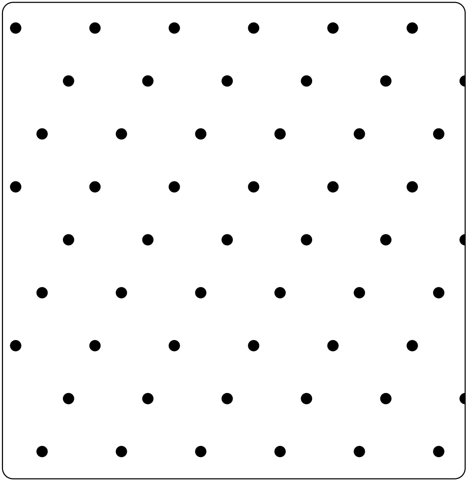
- while there exist i such that $\|b_i\|_2 > 4/3 \cdot \lambda_1(L_i)$
  (L_i is roughly the lattice spanned by (b_i, b_{i+1}))
  - run Lagrange-Gauss on L_i

This algorithm

- finds $v \in L$ with $\|v\|_2 \leq 2^n \cdot \lambda_1(L)$
- runs in polynomial time

[LLL82] Lenstra, Lenstra, and Lovász. Factoring polynomials with rational coefficients. Mathematische annalen.

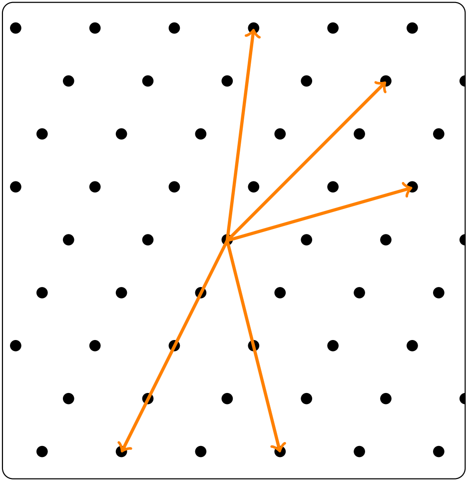
Sieving algorithm [AKS01]



Sieving:

[AKS01] Ajtai, Kumar, and Sivakumar. A sieve algorithm for the shortest lattice vector problem. STOC

Sieving algorithm [AKS01]

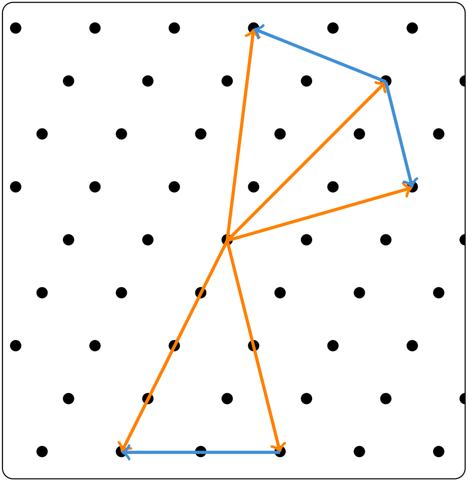


Sieving:

- Create many large vectors

[AKS01] Ajtai, Kumar, and Sivakumar. A sieve algorithm for the shortest lattice vector problem. STOC

Sieving algorithm [AKS01]

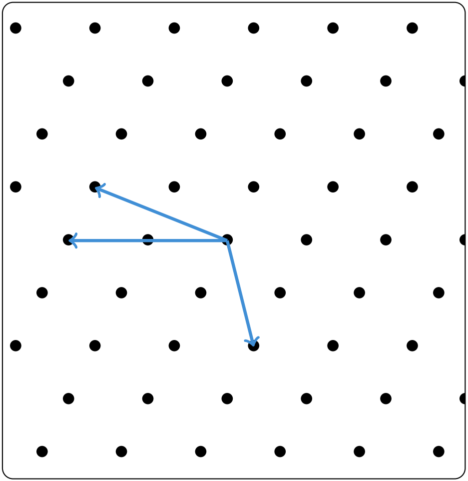


Sieving:

- Create many large vectors
- Subtract close ones to create shorter vectors

[AKS01] Ajtai, Kumar, and Sivakumar. A sieve algorithm for the shortest lattice vector problem. STOC

Sieving algorithm [AKS01]

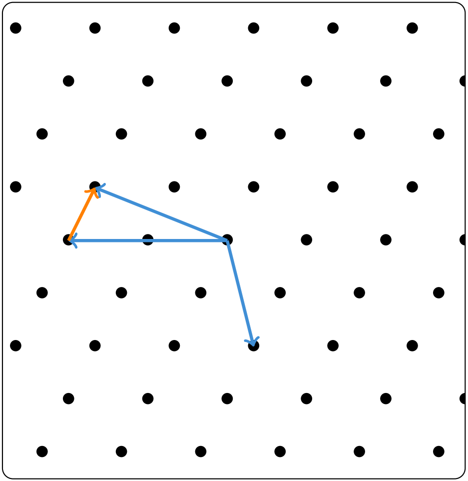


Sieving:

- Create many large vectors
- Subtract close ones to create shorter vectors
- Repeat with the shorter vectors

[AKS01] Ajtai, Kumar, and Sivakumar. A sieve algorithm for the shortest lattice vector problem. STOC

Sieving algorithm [AKS01]

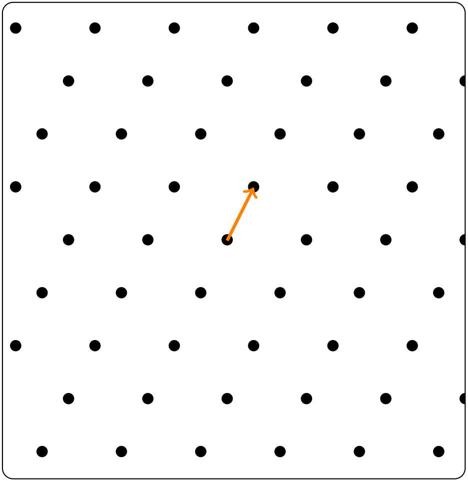


Sieving:

- Create many large vectors
- Subtract close ones to create shorter vectors
- Repeat with the shorter vectors

[AKS01] Ajtai, Kumar, and Sivakumar. A sieve algorithm for the shortest lattice vector problem. STOC

Sieving algorithm [AKS01]

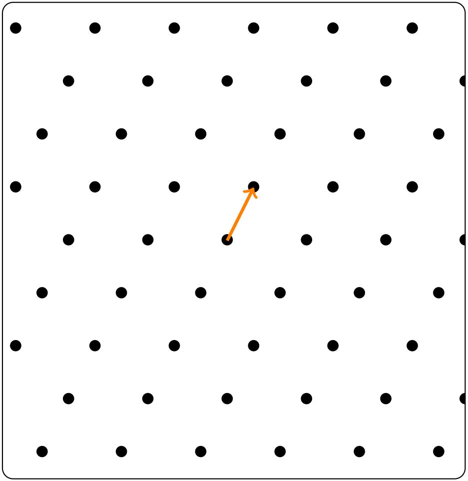


Sieving:

- Create many large vectors
- Subtract close ones to create shorter vectors
- Repeat with the shorter vectors

[AKS01] Ajtai, Kumar, and Sivakumar. A sieve algorithm for the shortest lattice vector problem. STOC

Sieving algorithm [AKS01]



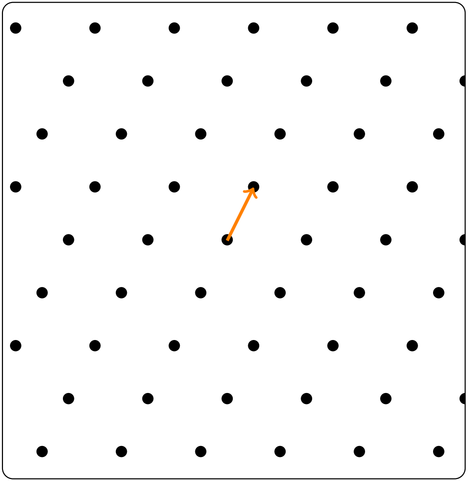
Sieving:

- Create many large vectors
- Subtract close ones to create shorter vectors
- Repeat with the shorter vectors

Size of the initial list: $2^{O(n)}$

[AKS01] Ajtai, Kumar, and Sivakumar. A sieve algorithm for the shortest lattice vector problem. STOC

Sieving algorithm [AKS01]



Sieving:

- Create many large vectors
- Subtract close ones to create shorter vectors
- Repeat with the shorter vectors

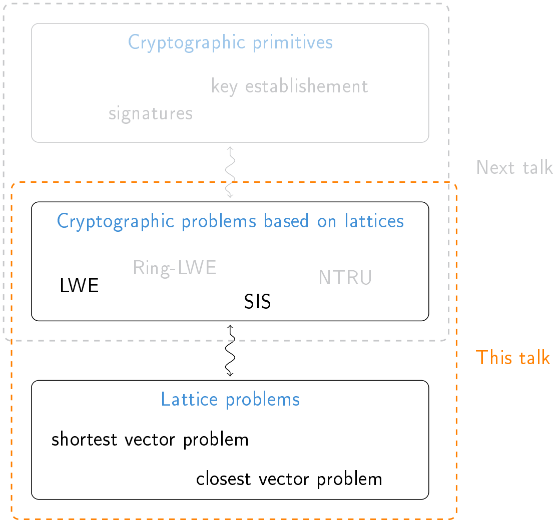
Size of the initial list: $2^{O(n)}$

- finds a shortest vector
- runs in time $2^{O(n)}$

[AKS01] Ajtai, Kumar, and Sivakumar. A sieve algorithm for the shortest lattice vector problem. STOC

Conclusion

What we have seen



Takeaway: all these problems are supposed to be quantumly hard

(for a good choice of parameters)