Public-Key Cryptosystems from the Worst-Case Shortest Vector Problem

Chris Peikert
SRI → Georgia Tech

Impagliazzo's World Workshop

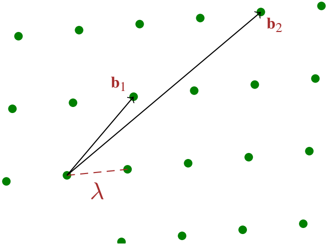
This Talk

1. State of Lattice-Based Cryptography
2. Main Result: Public-Key Encryption based on GapSVP
3. Proof & Future Work

Shortest Vector Problem(s)

A lattice $\mathcal{L} \subset \mathbb{R}^n$ having basis $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ is:

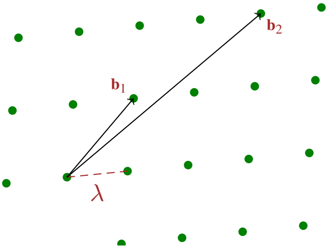
$$\mathcal{L} = \sum_{i=1}^{n} (\mathbb{Z} \cdot \mathbf{b}_i)$$



Shortest Vector Problem(s)

A lattice $\mathcal{L} \subset \mathbb{R}^n$ having basis $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ is:

$$\mathcal{L} = \sum_{i=1}^{n} (\mathbb{Z} \cdot \mathbf{b}_i)$$



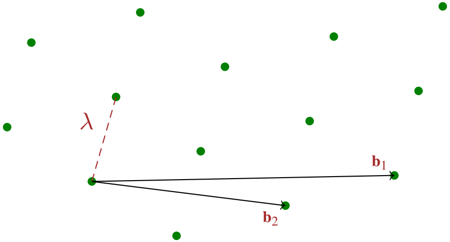
Shortest Vector Problem (γ-GapSVP)

- Given $\mathbf{B}$, decide: $\lambda \leq 1$ or $\lambda > \gamma$?

Shortest Vector Problem(s)

A lattice $\mathcal{L} \subset \mathbb{R}^n$ having basis $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ is:

$$\mathcal{L} = \sum_{i=1}^{n} (\mathbb{Z} \cdot \mathbf{b}_i)$$



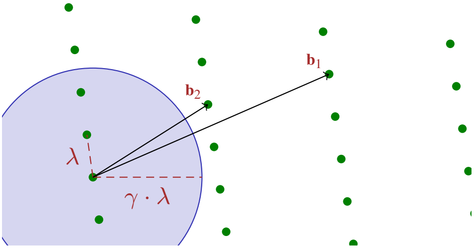
Shortest Vector Problem (γ-GapSVP)

- Given $\mathbf{B}$, decide: $\lambda \leq 1$ or $\lambda > \gamma$?

Shortest Vector Problem(s)

A lattice $\mathcal{L} \subset \mathbb{R}^n$ having basis $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ is:

$$\mathcal{L} = \sum_{i=1}^{n} (\mathbb{Z} \cdot \mathbf{b}_i)$$



Shortest Vector Problem (γ-GapSVP)

- Given $\mathbf{B}$, decide: $\lambda \leq 1$ or $\lambda > \gamma$?

Unique SVP (γ-uSVP)

- Given $\mathbf{B}$ with 'γ-unique' shortest vector, find it.

Worst-Case Complexity

GapSVP

$\gamma =$	$2^{(\log n)^{1-\epsilon}}$	$\sqrt{n}$	n	$2^{\sim n}$
	NP-hard*	$\in$ coNP	(some) crypto	$\in$ P
	[Ajt98,...,HR07]	[GG98,AR05]	[Ajt96,..., MR04,Reg05]	[LLL82,Sch87]

Worst-Case Complexity

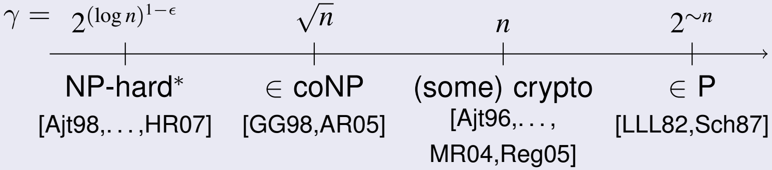
GapSVP

$\gamma =$	$2^{(\log n)^{1-\epsilon}}$	$\sqrt{n}$	n	$2^{\sim n}$
	NP-hard*	$\in$ coNP	(some) crypto	$\in$ P
	[Ajt98,...,HR07]	[GG98,AR05]	[Ajt96,..., MR04,Reg05]	[LLL82,Sch87]

- For $\gamma = \text{poly}(n)$, best algorithm is 2^n time & space [AKS01]

Worst-Case Complexity

GapSVP



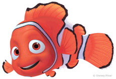
- For $\gamma = \text{poly}(n)$, best algorithm is 2^n time & space [AKS01]

uSVP

Taxonomy of Lattice-Based Crypto

'minicrypt'

OWF [Ajt96, . . .]

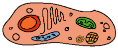


Sigs
[LM08,GPV08]

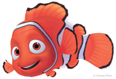
ID schemes
[MV03,Lyu08]

Taxonomy of Lattice-Based Crypto

'minicrypt'



OWF [Ajt96,. . .]



Sigs
[LM08,GPV08]

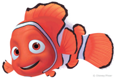
ID schemes
[MV03,Lyu08]

☞ GapSVP etc. hard

Taxonomy of Lattice-Based Crypto

'minicrypt'

OWF [Ajt96, . . .]



Sigs
[LM08,GPV08]

ID schemes
[MV03,Lyu08]

'CRYPTOMANIA'

PKE [AD97,Reg03,Reg05]

CCA [PW08]

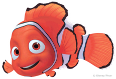
ID-based [GPV08]

☞ GapSVP etc. hard

Taxonomy of Lattice-Based Crypto

'minicrypt'

OWF [Ajt96,. . .]



Sigs
[LM08,GPV08]

ID schemes
[MV03,Lyu08]

☞ GapSVP etc. hard

'CRYPTOMANIA'

PKE [AD97,Reg03,Reg05]

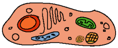
CCA [PW08]

ID-based [GPV08]

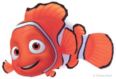
(Obl. tran. [PVW08], leakage [AGV09],
homom [G09], KDM [ACPS09], HIBE [P09])

Taxonomy of Lattice-Based Crypto

'minicrypt'



OWF [Ajt96, . . .]



Sigs
[LM08,GPV08]

ID schemes
[MV03,Lyu08]

☞ GapSVP etc. hard

'CRYPTOMANIA'

PKE [AD97,Reg03,Reg05]

CCA [PW08]

ID-based [GPV08]

(Obl. tran. [PVW08], leakage [AGV09],
homom [G09], KDM [ACPS09], HIBE [P09])

☞ uSVP hard

☞ GapSVP etc. *quantum*-hard

Learning With Errors

- Generalizes 'learning parity with noise': dim n, modulus $q \geq 2$

Learning With Errors

- Generalizes 'learning parity with noise': dim n, modulus $q \geq 2$
- **Search:** find $\mathbf{s} \in \mathbb{Z}_q^n$ given 'noisy random inner products'

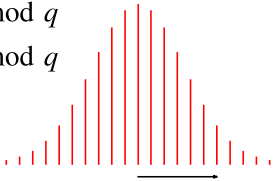
$$\begin{aligned} \mathbf{a}_1 \quad &, \quad b_1 \approx \langle \mathbf{a}_1 \, , \mathbf{s} \rangle \mod q \\ \mathbf{a}_2 \quad &, \quad b_2 \approx \langle \mathbf{a}_2 \, , \mathbf{s} \rangle \mod q \\ &\vdots \end{aligned}$$

Learning With Errors

- Generalizes 'learning parity with noise': dim n, modulus $q \geq 2$
- **Search:** find $\mathbf{s} \in \mathbb{Z}_q^n$ given 'noisy random inner products'

$$\begin{aligned} \mathbf{a}_1 \quad &, \quad b_1 = \langle \mathbf{a}_1 \, , \, \mathbf{s} \rangle + x_1 \mod q \\ \mathbf{a}_2 \quad &, \quad b_2 = \langle \mathbf{a}_2 \, , \, \mathbf{s} \rangle + x_2 \mod q \\ &\vdots \end{aligned}$$

Uniform $\mathbf{a}_i \in \mathbb{Z}_q^n$, Gaussian errors x_i



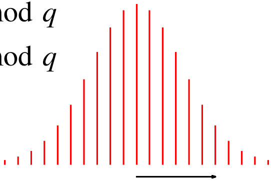
$\alpha \cdot q \geq \sqrt{n}$

Learning With Errors

- Generalizes 'learning parity with noise': dim n, modulus $q \geq 2$
- **Search:** find $\mathbf{s} \in \mathbb{Z}_q^n$ given 'noisy random inner products'

$$\mathbf{a}_1 \quad , \quad b_1 = \langle \mathbf{a}_1 \, , \, \mathbf{s} \rangle + x_1 \mod q$$
$$\mathbf{a}_2 \quad , \quad b_2 = \langle \mathbf{a}_2 \, , \, \mathbf{s} \rangle + x_2 \mod q$$
$$\vdots$$

Uniform $\mathbf{a}_i \in \mathbb{Z}_q^n$, Gaussian errors x_i



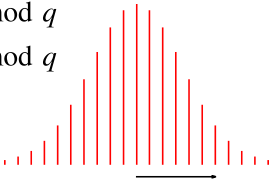
- **Decision:** distinguish from uniform $(\mathbf{a}_i \, , \, b_i)$

Learning With Errors

- Generalizes 'learning parity with noise': dim n, modulus $q \geq 2$
- **Search**: find $\mathbf{s} \in \mathbb{Z}_q^n$ given 'noisy random inner products'

$$\mathbf{a}_1 \quad , \quad b_1 = \langle \mathbf{a}_1 \, , \, \mathbf{s} \rangle + x_1 \mod q$$
$$\mathbf{a}_2 \quad , \quad b_2 = \langle \mathbf{a}_2 \, , \, \mathbf{s} \rangle + x_2 \mod q$$
$$\vdots$$

Uniform $\mathbf{a}_i \in \mathbb{Z}_q^n$, Gaussian errors x_i



$\alpha \cdot q \geq \sqrt{n}$

- **Decision**: distinguish from uniform $(\mathbf{a}_i \, , \, b_i)$

State of the Art

(n/α)-GapSVP etc. $\leq$ search-LWE $\leq$ decision-LWE $\leq$ crypto

- **quantum** [Reg05]
- prime $q = \text{poly}(n)$ [BFKL94,R05]
- [R05,PW08,GPV08, PVW08,AGV09,ACPS09,...]

Our Results

First public-key encryption based on classical GapSVP hardness

Our Results

First public-key encryption based on classical GapSVP hardness

1. **Classical reduction: GapSVP $\leq$ Learning With Errors**

Our Results

First public-key encryption based on classical GapSVP hardness

1. **Classical reduction: GapSVP $\leq$ Learning With Errors**
   - ⋆ Standard (n/α)-GapSVP: large LWE modulus $q \geq 2^n$

Our Results

First public-key encryption based on classical GapSVP hardness

1. **Classical reduction: GapSVP $\leq$ Learning With Errors**
   - ⋆ Standard (n/α)-GapSVP: large LWE modulus $q \geq 2^n$
   - ⋆ 'Improve ζ to (n/α)'-GapSVP: $q \approx \zeta$ [$=$ poly(n)]

Our Results

First public-key encryption based on classical GapSVP hardness

1. **Classical reduction: GapSVP $\leq$ Learning With Errors**
   - ⋆ Standard (n/α)-GapSVP: large LWE modulus $q \geq 2^n$
   - ⋆ 'Improve ζ to (n/α)'-GapSVP: $q \approx \zeta$ [$=$ poly(n)]

2. **LWE search = decision for large q** [$\gg$ poly(n)]
   - $\Rightarrow$ GapSVP-hardness of prior LWE-based crypto [Reg05,...]

Our Results

First public-key encryption based on classical GapSVP hardness

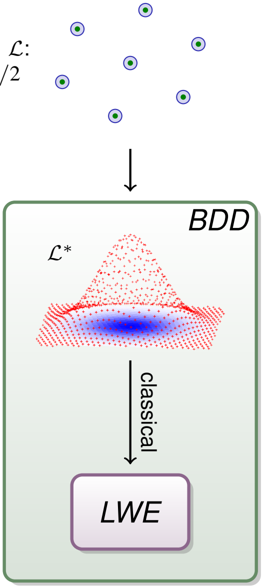
1. **Classical reduction: GapSVP $\leq$ Learning With Errors**
   - ⋆ Standard (n/α)-GapSVP: large LWE modulus $q \geq 2^n$
   - ⋆ 'Improve ζ to (n/α)'-GapSVP: $q \approx \zeta$ [$=$ poly(n)]

2. **LWE search = decision for large q** [$\gg$ poly(n)]
   - $\Rightarrow$ GapSVP-hardness of prior LWE-based crypto [Reg05,...]

3. **New LWE-based chosen ciphertext-secure encryption**
   - ⋆ Much simpler, milder assumption than prior CCA [PW08]

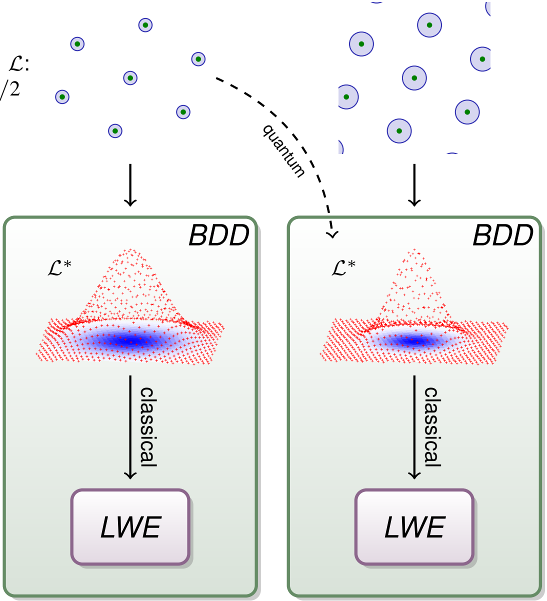
[Regev05] Reduction to LWE

BDD on $\mathcal{L}$:
$d \ll \lambda/2$

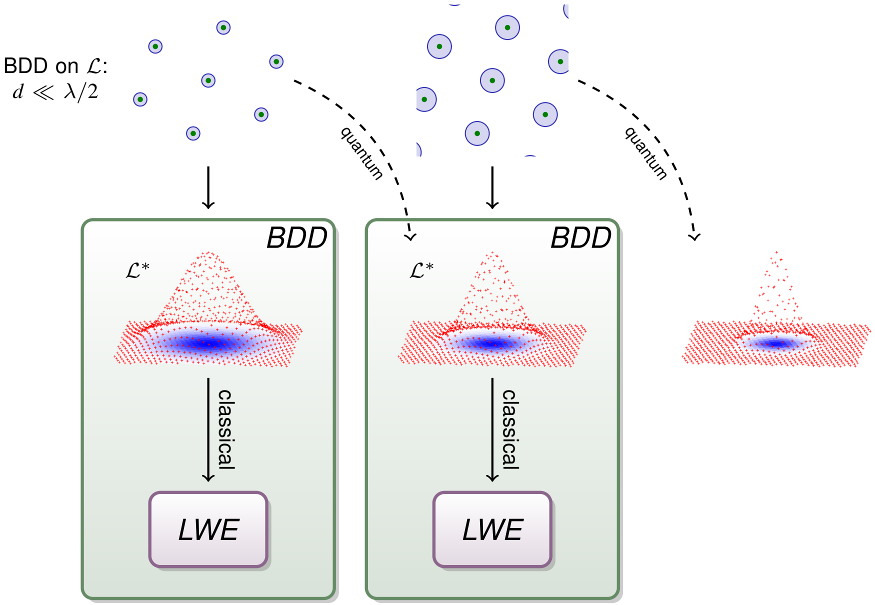


[Regev05] Reduction to LWE

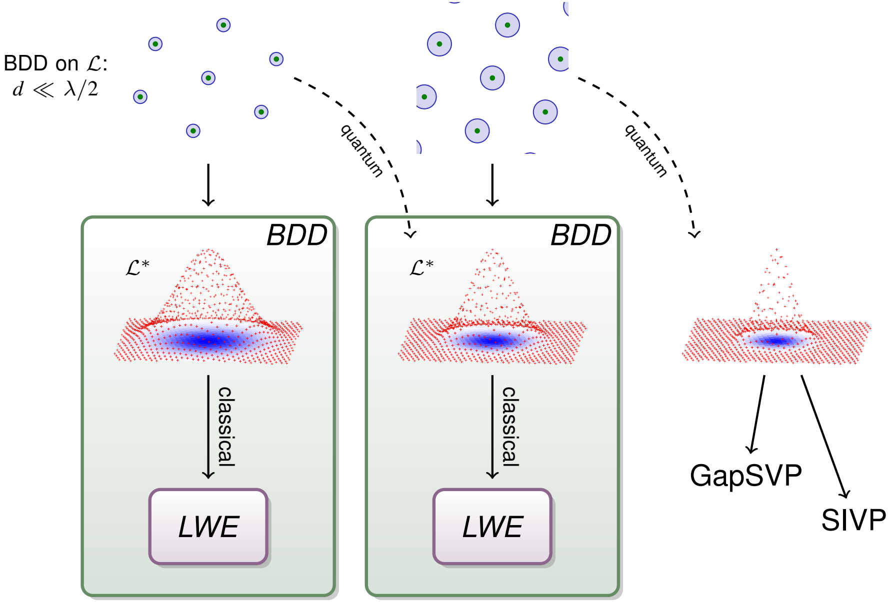
BDD on $\mathcal{L}$:
$d \ll \lambda/2$



[Regev05] Reduction to LWE

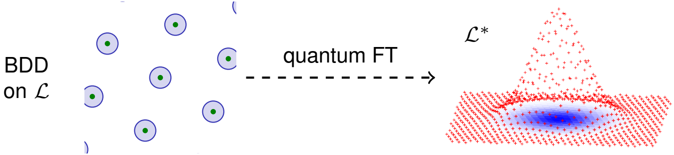


[Regev05] Reduction to LWE



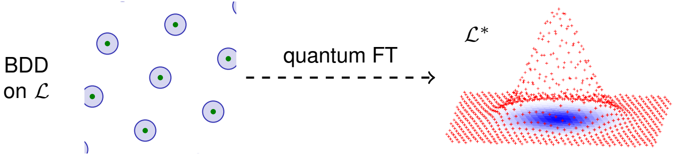
Why Quantum?

- “Obvious” answer: iterative step



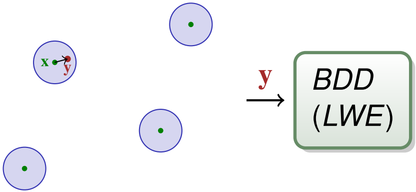
Why Quantum?

- “Obvious” answer: iterative step



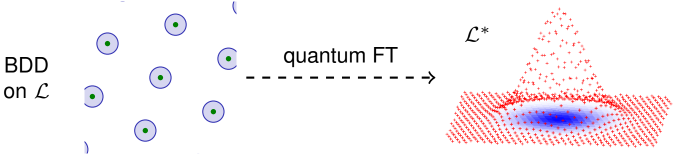
- Another answer: to make use of BDD/LWE oracle

1. Choose some $\mathbf{x} \in \mathcal{L}$
2. Perturb to $\mathbf{y} \approx \mathbf{x}$
3. Invoke oracle on $\mathbf{y}$



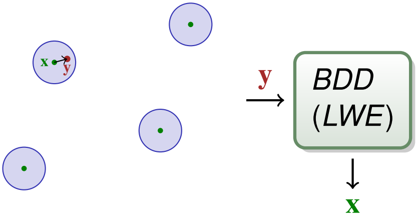
Why Quantum?

- “Obvious” answer: iterative step



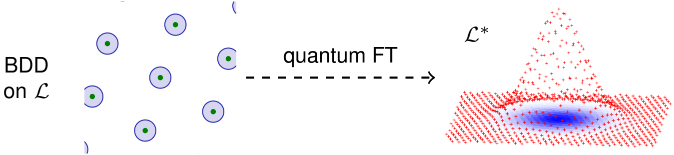
- Another answer: to make use of BDD/LWE oracle

1. Choose some $\mathbf{x} \in \mathcal{L}$
2. Perturb to $\mathbf{y} \approx \mathbf{x}$
3. Invoke oracle on $\mathbf{y}$
4. Returns $\mathbf{x}$ —
   we already knew that!



Why Quantum?

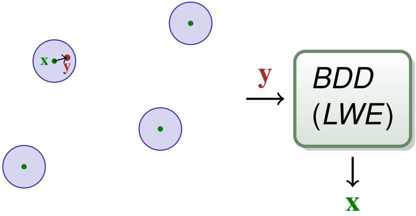
- "Obvious" answer: iterative step



- Another answer: to make use of BDD/LWE oracle

1. Choose some $\mathbf{x} \in \mathcal{L}$
2. Perturb to $\mathbf{y} \approx \mathbf{x}$
3. Invoke oracle on $\mathbf{y}$
4. Returns $\mathbf{x}$ — we already knew that!

✔ *Quantum can "uncompute"* $\mathbf{x}$



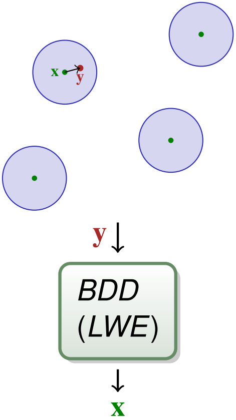
Our Approach

New way of solving GapSVP in a reduction

Our Approach

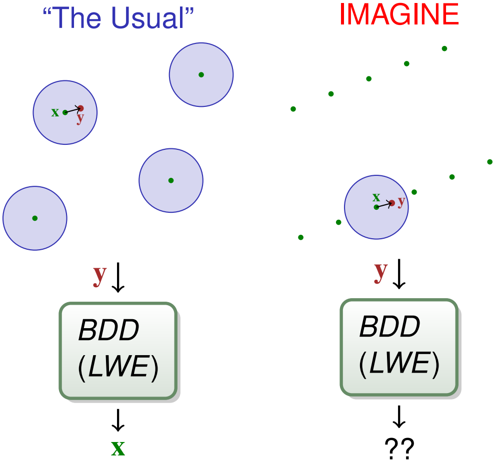
New way of solving GapSVP in a reduction

"The Usual"



Our Approach

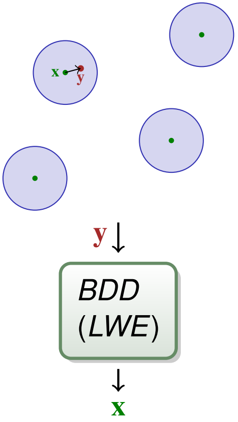
New way of solving GapSVP in a reduction



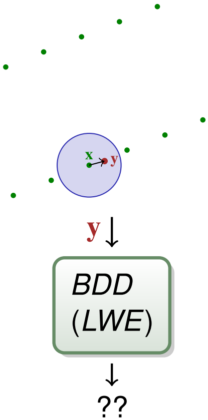
Our Approach

New way of solving GapSVP in a reduction

"The Usual"



IMAGINE



Illegal BDD instance

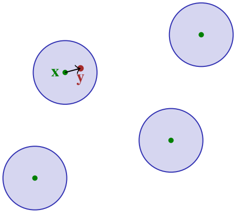
⇓

Incorrect (& unknown!) LWE distribution

Our Approach

New way of solving GapSVP in a reduction

"The Usual"

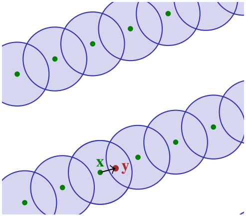


$\mathbf{y} \downarrow$

BDD
(*LWE*)

$\downarrow$
$\mathbf{x}$

IMAGINE



$\mathbf{y} \downarrow$

BDD
(*LWE*)

$\downarrow$
??

Illegal BDD instance
$\Downarrow$
Incorrect (& unknown!)
LWE distribution

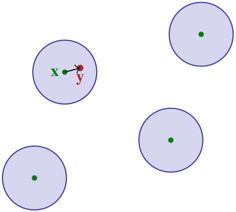
SO WHAT!

When $\lambda \ll d$,
oracle cannot guess $\mathbf{x}$
$\Downarrow$
Distinguishes large λ
from small

Our Approach

New way of solving GapSVP in a reduction

"The Usual"



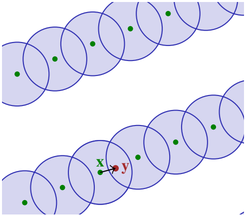
$\mathbf{y}\downarrow$

BDD (*LWE*)

$\downarrow$

$\mathbf{x}$

IMAGINE



$\mathbf{y}\downarrow$

BDD (*LWE*)

$\downarrow$

??

Illegal BDD instance

$\Downarrow$

Incorrect (& unknown!) LWE distribution

SO WHAT!

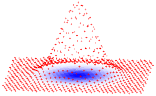
When $\lambda \ll d$,
oracle cannot guess $\mathbf{x}$

$\Downarrow$

Distinguishes large λ from small

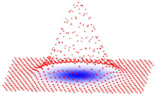
- View as [GoldGold98] AM proof between reduction and oracle

Technical Obstacles

1. What about  in $BDD \rightarrow LWE$ reduction?

(No quantum allowed!)

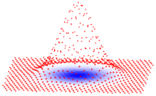
Technical Obstacles

1. What about  in *BDD* $\rightarrow$ *LWE* reduction?

   (No quantum allowed!)

   ⋆ Use [GPV08] sampling algorithm with 'best available' basis for $\mathcal{L}^*$.

Technical Obstacles

1. What about  in *BDD* $\longrightarrow$ *LWE* reduction?

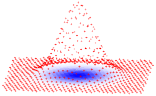
   (No quantum allowed!)

   ★ Use [GPV08] sampling algorithm with 'best available' basis for $\mathcal{L}^*$.

   'ζ-good' basis $\Rightarrow$ LWE modulus $q \approx \zeta$.

   (LLL-reduced basis is 2^n-good.)

Technical Obstacles

1. What about  in *BDD* $\longrightarrow$ *LWE* reduction?

(No quantum allowed!)

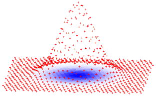
- ⋆ Use [GPV08] sampling algorithm with 'best available' basis for $\mathcal{L}^*$.

'ζ-good' basis $\Rightarrow$ LWE modulus $q \approx \zeta$.

(LLL-reduced basis is 2^n-good.)

- ⋆ 'One shot' (non-iterative) reduction

Technical Obstacles

1. What about  in *BDD* $\longrightarrow$ *LWE* reduction?

   (No quantum allowed!)

   ⋆ Use [GPV08] sampling algorithm with 'best available' basis for $\mathcal{L}^*$.

   'ζ-good' basis $\Rightarrow$ LWE modulus $q \approx \zeta$.

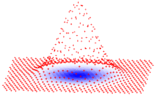
   (LLL-reduced basis is 2^n-good.)

   ⋆ 'One shot' (non-iterative) reduction

2. LWE search / decision equivalence?

   (Normally requires prime $q = \text{poly}(n) \dots$)

Technical Obstacles

1. What about  in *BDD* $\to$ *LWE* reduction?

(No quantum allowed!)

- ⋆ Use [GPV08] sampling algorithm with 'best available' basis for $\mathcal{L}^*$.

'ζ-good' basis $\Rightarrow$ LWE modulus $q \approx \zeta$.

(LLL-reduced basis is 2^n-good.)

- ⋆ 'One shot' (non-iterative) reduction

2. LWE search / decision equivalence?

(Normally requires prime $q = \text{poly}(n) \ldots$)

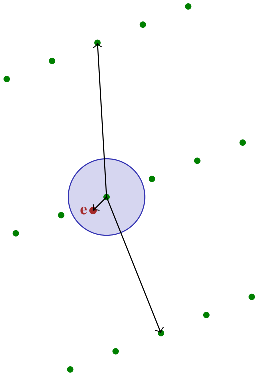
Option 1: crypto directly based on search-LWE

Option 2: search = decision for 'smooth' q and Gaussian error

Details of Reduction

Given any ("ζ-good") $\mathbf{B}$:

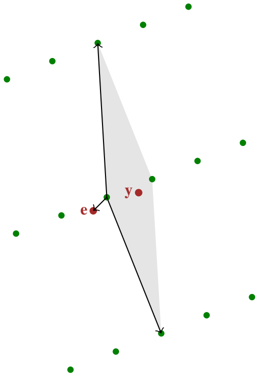
1. Choose $\mathbf{e} \leftarrow \sqrt{n} \cdot \mathcal{B}_n$



Details of Reduction

Given any ("ζ-good") $\mathbf{B}$:

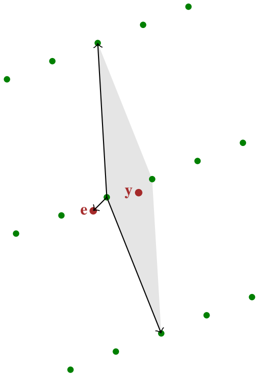
1. Choose $\mathbf{e} \leftarrow \sqrt{n} \cdot \mathcal{B}_n$
2. Let $\mathbf{y} = \mathbf{e} \bmod \mathbf{B}$



Details of Reduction

Given any ("ζ-good") $\mathbf{B}$:

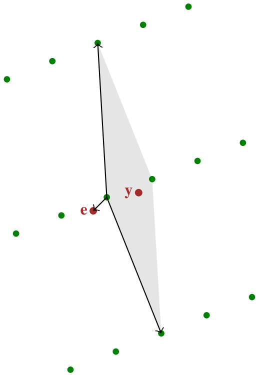
1. Choose $\mathbf{e} \leftarrow \sqrt{n} \cdot \mathcal{B}_n$
2. Let $\mathbf{y} = \mathbf{e} \bmod \mathbf{B}$
3. (Get some $\mathbf{x} \in \mathcal{L}$ from LWE oracle somehow. . .)



Details of Reduction

Given any ("ζ-good") $\mathbf{B}$:

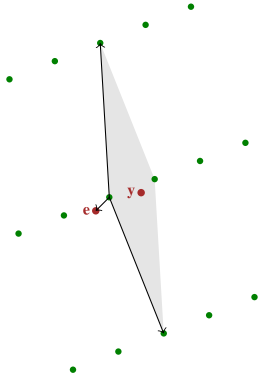
1. Choose $\mathbf{e} \leftarrow \sqrt{n} \cdot \mathcal{B}_n$
2. Let $\mathbf{y} = \mathbf{e} \bmod \mathbf{B}$
3. (Get some $\mathbf{x} \in \mathcal{L}$ from LWE oracle somehow...)
4. If $\mathbf{y} - \mathbf{x} = \mathbf{e}$, output "large," else output "small"



Details of Reduction

Given any (“ζ-good”) $\mathbf{B}$:

1. Choose $\mathbf{e} \leftarrow \sqrt{n} \cdot \mathcal{B}_n$
2. Let $\mathbf{y} = \mathbf{e} \bmod \mathbf{B}$
3. (Get some $\mathbf{x} \in \mathcal{L}$ from LWE oracle somehow. . .)
4. If $\mathbf{y} - \mathbf{x} = \mathbf{e}$, output “large,” else output “small”



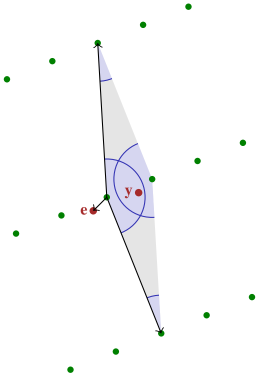
Analysis for $\lambda \leq 1$:

Let $\mathbf{0} \neq \mathbf{v} \in \mathcal{L}$ be shortest.

Details of Reduction

Given any (“ζ-good”) $\mathbf{B}$:

1. Choose $\mathbf{e} \leftarrow \sqrt{n} \cdot \mathcal{B}_n$
2. Let $\mathbf{y} = \mathbf{e} \bmod \mathbf{B}$
3. (Get some $\mathbf{x} \in \mathcal{L}$ from LWE oracle somehow. . .)
4. If $\mathbf{y} - \mathbf{x} = \mathbf{e}$, output “large,” else output “small”



Analysis for $\lambda \leq 1$:

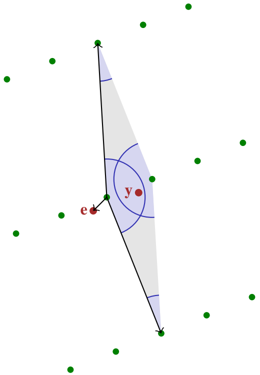
Let $\mathbf{0} \neq \mathbf{v} \in \mathcal{L}$ be shortest.

$(\sqrt{n} \cdot \mathcal{B}_n) \cap (\mathbf{v} + \sqrt{n} \cdot \mathcal{B}_n)$ is a noticeable fraction of $\sqrt{n} \cdot \mathcal{B}_n$.

Details of Reduction

Given any ("ζ-good") $\mathbf{B}$:

1. Choose $\mathbf{e} \leftarrow \sqrt{n} \cdot \mathcal{B}_n$
2. Let $\mathbf{y} = \mathbf{e} \bmod \mathbf{B}$
3. (Get some $\mathbf{x} \in \mathcal{L}$ from LWE oracle somehow. . .)
4. If $\mathbf{y} - \mathbf{x} = \mathbf{e}$, output "large," else output "small"



Analysis for $\lambda \leq 1$:

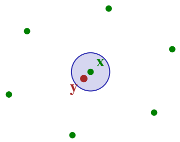
Let $\mathbf{0} \neq \mathbf{v} \in \mathcal{L}$ be shortest.

$(\sqrt{n} \cdot \mathcal{B}_n) \cap (\mathbf{v} + \sqrt{n} \cdot \mathcal{B}_n)$ is a noticeable fraction of $\sqrt{n} \cdot \mathcal{B}_n$.

$\Rightarrow$ Step 3 (no matter what it is!) can't guess original $\mathbf{e}$.

Reduction: Step 3

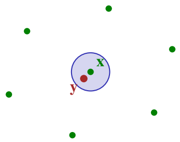
Given "ζ-good" $\mathbf{B}$ and $\mathbf{y} = \mathbf{x} + \mathbf{e}$ for $\mathbf{x} = \mathbf{B}\mathbf{c} \in \mathcal{L}$ and $\|\mathbf{e}\| \leq \sqrt{n}$.



Reduction: Step 3

Given "ζ-good" $\mathbf{B}$ and $\mathbf{y} = \mathbf{x} + \mathbf{e}$ for $\mathbf{x} = \mathbf{Bc} \in \mathcal{L}$ and $\|\mathbf{e}\| \leq \sqrt{n}$.

To generate sample $(\mathbf{a}, b)$ from $A_{\mathbf{s},\alpha}$ for $\mathbf{s} = \mathbf{c} \bmod q$ and $q = \zeta \cdot (\sqrt{n}/\alpha)$:

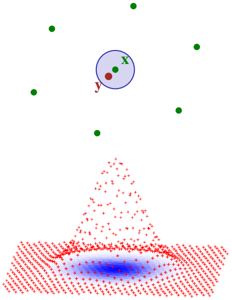


Reduction: Step 3

Given "ζ-good" $\mathbf{B}$ and $\mathbf{y} = \mathbf{x} + \mathbf{e}$ for $\mathbf{x} = \mathbf{B}\mathbf{c} \in \mathcal{L}$ and $\|\mathbf{e}\| \leq \sqrt{n}$.

To generate sample $(\mathbf{a}, b)$ from $A_{\mathbf{s},\alpha}$ for $\mathbf{s} = \mathbf{c} \bmod q$ and $q = \zeta \cdot (\sqrt{n}/\alpha)$:

1. Using $\mathbf{B}^* = \mathbf{B}^{-t}$, sample $\mathbf{z} \leftarrow D_{\mathcal{L}^*,\zeta}$ using [GPV08]



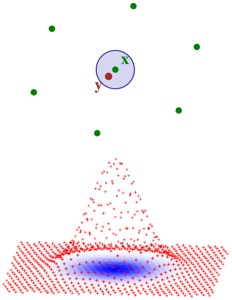
Reduction: Step 3

Given "ζ-good" $\mathbf{B}$ and $\mathbf{y} = \mathbf{x} + \mathbf{e}$ for $\mathbf{x} = \mathbf{Bc} \in \mathcal{L}$ and $\|\mathbf{e}\| \leq \sqrt{n}$.

To generate sample $(\mathbf{a}, b)$ from $A_{\mathbf{s},\alpha}$ for $\mathbf{s} = \mathbf{c} \bmod q$ and $q = \zeta \cdot (\sqrt{n}/\alpha)$:

1. Using $\mathbf{B}^* = \mathbf{B}^{-t}$, sample $\mathbf{z} \leftarrow D_{\mathcal{L}^*,\zeta}$ using [GPV08]
2. Write $\mathbf{v} = \mathbf{B}^*\mathbf{z}$ for $\mathbf{z} \in \mathbb{Z}^n$. Output

$$\mathbf{a} = \mathbf{z} \bmod q \quad \text{and} \quad b \simeq \langle \mathbf{v}, \mathbf{y} \rangle \bmod q$$



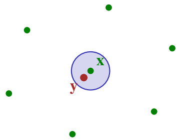
Reduction: Step 3

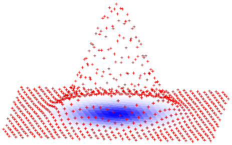
Given "ζ-good" $\mathbf{B}$ and $\mathbf{y} = \mathbf{x} + \mathbf{e}$ for $\mathbf{x} = \mathbf{Bc} \in \mathcal{L}$ and $\|\mathbf{e}\| \leq \sqrt{n}$.

To generate sample $(\mathbf{a}, b)$ from $A_{\mathbf{s},\alpha}$ for $\mathbf{s} = \mathbf{c} \bmod q$ and $q = \zeta \cdot (\sqrt{n}/\alpha)$:

1. Using $\mathbf{B}^* = \mathbf{B}^{-t}$, sample $\mathbf{z} \leftarrow D_{\mathcal{L}^*,\zeta}$ using [GPV08]
2. Write $\mathbf{v} = \mathbf{B}^*\mathbf{z}$ for $\mathbf{z} \in \mathbb{Z}^n$. Output

$$\mathbf{a} = \mathbf{z} \bmod q \quad \text{and} \quad b \simeq \langle \mathbf{v}, \mathbf{y} \rangle \bmod q$$





Analysis for $\lambda > n/\alpha$:

- $\zeta \geq q \cdot (\sqrt{n}/\lambda) \Rightarrow$ uniform $\mathbf{a} \in \mathbb{Z}_q^n$. [MR04]

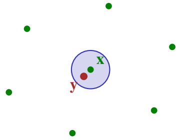
Reduction: Step 3

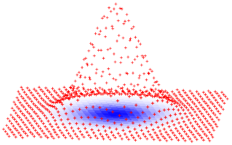
Given "ζ-good" $\mathbf{B}$ and $\mathbf{y} = \mathbf{x} + \mathbf{e}$ for $\mathbf{x} = \mathbf{Bc} \in \mathcal{L}$ and $\|\mathbf{e}\| \le \sqrt{n}$.

To generate sample $(\mathbf{a}, b)$ from $A_{\mathbf{s},\alpha}$ for $\mathbf{s} = \mathbf{c} \bmod q$ and $q = \zeta \cdot (\sqrt{n}/\alpha)$:

1. Using $\mathbf{B}^* = \mathbf{B}^{-t}$, sample $\mathbf{z} \leftarrow D_{\mathcal{L}^*,\zeta}$ using [GPV08]
2. Write $\mathbf{v} = \mathbf{B}^*\mathbf{z}$ for $\mathbf{z} \in \mathbb{Z}^n$. Output

$$\mathbf{a} = \mathbf{z} \bmod q \quad \text{and} \quad b \simeq \langle \mathbf{v}, \mathbf{y} \rangle \bmod q$$





Analysis for $\lambda > n/\alpha$:

- $\zeta \ge q \cdot (\sqrt{n}/\lambda) \Rightarrow$ uniform $\mathbf{a} \in \mathbb{Z}_q^n$. [MR04]
- Condition on $\mathbf{a}$. Then $b = \langle \mathbf{v}, \mathbf{x} + \mathbf{e} \rangle$

$$= \langle \mathbf{B}^*\mathbf{z}, \mathbf{Bc} \rangle + \langle \mathbf{v}, \mathbf{e} \rangle \simeq \langle \mathbf{a}, \mathbf{s} \rangle + D_{\zeta \cdot \|\mathbf{e}\|} \bmod q.$$

Finally, $\zeta \cdot \|\mathbf{e}\| \le \alpha \cdot q$.

Reducing Search to Decision

- Suppose D distinguishes $(\mathbf{a} \in \mathbb{Z}_q^n \ , \ b \approx \langle \mathbf{a}, \mathbf{s} \rangle) \leftarrow A_{\mathbf{s},\alpha}$ from uniform.

Reducing Search to Decision

- Suppose D distinguishes $(\mathbf{a} \in \mathbb{Z}_q^n\ ,\ b \approx \langle \mathbf{a}, \mathbf{s} \rangle) \leftarrow A_{\mathbf{s},\alpha}$ from uniform.
- Let $q = q_1 \cdots q_t$ [$\gg$ poly(n)] for distinct $(1/\alpha) \leq q_i \leq$ poly(n).

Reducing Search to Decision

- Suppose D distinguishes $(\mathbf{a} \in \mathbb{Z}_q^n \ , \ b \approx \langle \mathbf{a}, \mathbf{s} \rangle) \leftarrow A_{\mathbf{s},\alpha}$ from uniform.
- Let $q = q_1 \cdots q_t$ [$\gg$ poly(n)] for distinct $(1/\alpha) \leq q_i \leq$ poly(n).

Find $\mathbf{s}$: Chinese remaindering & "smoothing"

- To test if $s_1 = 0 \bmod q_i$:

$$(\mathbf{a} \ , \ b) \mapsto (\mathbf{a} + r \cdot \mathbf{e}_1 \ , \ b) \quad \text{for} \quad r \leftarrow (q/q_i) \cdot \mathbb{Z}_{q_i}$$

Reducing Search to Decision

- Suppose D distinguishes $(\mathbf{a} \in \mathbb{Z}_q^n\ ,\ b \approx \langle \mathbf{a}, \mathbf{s} \rangle) \leftarrow A_{\mathbf{s},\alpha}$ from uniform.
- Let $q = q_1 \cdots q_t$ [$\gg$ poly(n)] for distinct $(1/\alpha) \leq q_i \leq$ poly(n).

Find $\mathbf{s}$: Chinese remaindering & "smoothing"

- To test if $s_1 = 0 \bmod q_i$:

$$(\mathbf{a}\ ,\ b) \mapsto (\mathbf{a} + r \cdot \mathbf{e}_1\ ,\ b) \quad \text{for} \quad r \leftarrow (q/q_i) \cdot \mathbb{Z}_{q_i}$$

- If yes, maps $A_{\mathbf{s},\alpha}$ to itself. If not, maps $A_{\mathbf{s},\alpha}$ to uniform ?

Reducing Search to Decision

- Suppose D distinguishes $(\mathbf{a} \in \mathbb{Z}_q^n \ , \ b \approx \langle \mathbf{a}, \mathbf{s} \rangle) \leftarrow A_{\mathbf{s},\alpha}$ from uniform.
- Let $q = q_1 \cdots q_t$ [$\gg$ poly(n)] for distinct $(1/\alpha) \leq q_i \leq$ poly(n).

Find $\mathbf{s}$: Chinese remaindering & "smoothing"

- To test if $s_1 = 0 \bmod q_i$:

$$(\mathbf{a} \ , \ b) \mapsto (\mathbf{a} + r \cdot \mathbf{e}_1 \ , \ b) \quad \text{for} \quad r \leftarrow (q/q_i) \cdot \mathbb{Z}_{q_i}$$

- If yes, maps $A_{\mathbf{s},\alpha}$ to itself. If not, maps $A_{\mathbf{s},\alpha}$ to uniform !

  Gaussians of width $\alpha q \geq (q/q_i)$ separated by (q/q_i)

  $\Rightarrow$ uniform* by smoothing bounds [MicReg04]

Reducing Search to Decision

- Suppose D distinguishes $(\mathbf{a} \in \mathbb{Z}_q^n\ ,\ b \approx \langle \mathbf{a}, \mathbf{s} \rangle) \leftarrow A_{\mathbf{s},\alpha}$ from uniform.
- Let $q = q_1 \cdots q_t$ [$\gg$ poly(n)] for distinct $(1/\alpha) \leq q_i \leq$ poly(n).

Find s: Chinese remaindering & "smoothing"

- To test if $s_1 = 0 \bmod q_i$:

$$(\mathbf{a}\ ,\ b) \mapsto (\mathbf{a} + r \cdot \mathbf{e}_1\ ,\ b) \quad \text{for} \quad r \leftarrow (q/q_i) \cdot \mathbb{Z}_{q_i}$$

- If yes, maps $A_{\mathbf{s},\alpha}$ to itself. If not, maps $A_{\mathbf{s},\alpha}$ to uniform !

  Gaussians of width $\alpha q \geq (q/q_i)$ separated by (q/q_i)

  $\Rightarrow$ uniform* by smoothing bounds [MicReg04]

- (NB: for general error dists, hybrid argument over q_i's fails.)

Chosen-Ciphertext Security

Intuitive Definition [RS91,DDN91,NY95]

- Encryption conceals message, even given decryption oracle

Chosen-Ciphertext Security

Intuitive Definition [RS91,DDN91,NY95]

- Encryption conceals message, even given decryption oracle

Elementary Construction

- Follows "witness-recovering decryption" approach [PW08].

Chosen-Ciphertext Security

Intuitive Definition [RS91,DDN91,NY95]

- Encryption conceals message, even given decryption oracle

Elementary Construction

- Follows "witness-recovering decryption" approach [PW08].
- Define $g_{\mathbf{A}}(\mathbf{s}, \mathbf{x}) = \mathbf{A}^t\mathbf{s} + \mathbf{x}$.
  Can generate $\mathbf{A}$ with "trapdoor" for $g_{\mathbf{A}}^{-1}$ [GGH97,Ajt99,AP09]

Chosen-Ciphertext Security

Intuitive Definition [RS91,DDN91,NY95]

- Encryption conceals message, even given decryption oracle

Elementary Construction

- Follows "witness-recovering decryption" approach [PW08].
- Define $g_{\mathbf{A}}(\mathbf{s}, \mathbf{x}) = \mathbf{A}^t \mathbf{s} + \mathbf{x}$.
  Can generate $\mathbf{A}$ with "trapdoor" for $g_{\mathbf{A}}^{-1}$ [GGH97,Ajt99,AP09]
- Distinguish $g_{\mathbf{A}_1}(\mathbf{s}, \mathbf{x}_1), \ldots, g_{\mathbf{A}_k}(\mathbf{s}, \mathbf{x}_k)$ [same $\mathbf{s}$!] $\iff$ solve LWE
  So $g_{\mathbf{A}_1}, \ldots, g_{\mathbf{A}_k}$ pseudorandom under 'correlated inputs' [RS09]

Chosen-Ciphertext Security

Intuitive Definition [RS91,DDN91,NY95]

- Encryption conceals message, even given decryption oracle

Elementary Construction

- Follows "witness-recovering decryption" approach [PW08].
- Define $g_{\mathbf{A}}(\mathbf{s}, \mathbf{x}) = \mathbf{A}^t \mathbf{s} + \mathbf{x}$.
  Can generate $\mathbf{A}$ with "trapdoor" for $g_{\mathbf{A}}^{-1}$ [GGH97,Ajt99,AP09]
- Distinguish $g_{\mathbf{A}_1}(\mathbf{s}, \mathbf{x}_1), \ldots, g_{\mathbf{A}_k}(\mathbf{s}, \mathbf{x}_k)$ [same $\mathbf{s}$!] $\iff$ solve LWE
  So $g_{\mathbf{A}_1}, \ldots, g_{\mathbf{A}_k}$ pseudorandom under 'correlated inputs' [RS09]
- Correlation-secure injective TDF $\Rightarrow$ CCA-secure encryption
  But much care needed to make $g_{\mathbf{A}}$ "chosen-output secure."

Epilogue

1. Using our main approach & other ideas, [LyuMic09] showed

$$\boxed{(\gamma\sqrt{n})\text{-GapSVP} \ \leq \ \gamma\text{-uSVP}} \ \leq \ \text{crypto}$$ [AjtaiDwork97,Regev03]

Epilogue

1. Using our main approach & other ideas, [LyuMic09] showed

$$\boxed{(\gamma\sqrt{n})\text{-GapSVP} \;\leq\; \gamma\text{-uSVP}} \;\leq\; \text{crypto} \quad \text{[AjtaiDwork97,Regev03]}$$

"Unifies" two styles of cryptosystems [AD97,Reg03] and [Reg05,. . .] under (almost) same assumption.

Epilogue

1. Using our main approach & other ideas, [LyuMic09] showed

$$\boxed{(\gamma\sqrt{n})\text{-GapSVP} \;\leq\; \gamma\text{-uSVP}} \;\leq\; \text{crypto} \quad \text{[AjtaiDwork97,Regev03]}$$

   "Unifies" two styles of cryptosystems [AD97,Reg03] and [Reg05,. . .] under (almost) same assumption.

2. **Open**: classical, iterative reduction to LWE

   Ought to solve GapSVP, SIVP, etc. for small $q = \text{poly}(n)$

Epilogue

1. Using our main approach & other ideas, [LyuMic09] showed

   $\boxed{(\gamma\sqrt{n})\text{-GapSVP} \ \leq \ \gamma\text{-uSVP}} \ \leq$ crypto [AjtaiDwork97,Regev03]

   "Unifies" two styles of cryptosystems [AD97,Reg03] and [Reg05,...] under (almost) same assumption.

2. **Open**: classical, iterative reduction to LWE

   Ought to solve GapSVP, SIVP, etc. for small $q = \text{poly}(n)$

3. **Open**: complexity of 'Improve ζ to γ'-GapSVP?

   NP-hard for nontrivial ζ? Better algorithms?