# On the Fine-Grained Hardness of Lattice Problems

Noah
Stephens-Davidowitz

Huck
Bennett

Alexander
Golovnev

Divesh Aggarwal

# Game Plan

# Game Plan

- Motivation

# Game Plan

- Motivation
  - How secure is lattice-based crypto?

# Game Plan

- Motivation
    - How secure is lattice-based crypto?
    - How sure are we?

# Game Plan

- Motivation
    - How secure is lattice-based crypto?
    - How sure are we?

- Summary of results

# Game Plan

- Motivation
    - How secure is lattice-based crypto?
    - How sure are we?

- Summary of results

- Fine-grained hardness of CVP

# Game Plan

- Motivation
    - How secure is lattice-based crypto?
    - How sure are we?

- Summary of results

- Fine-grained hardness of CVP

- Fine-grained hardness of SVP

# Game Plan

- Motivation
  - How secure is lattice-based crypto?
  - How sure are we?
- Summary of results
- Fine-grained hardness of CVP
- Fine-grained hardness of SVP
- Where do we go from here?

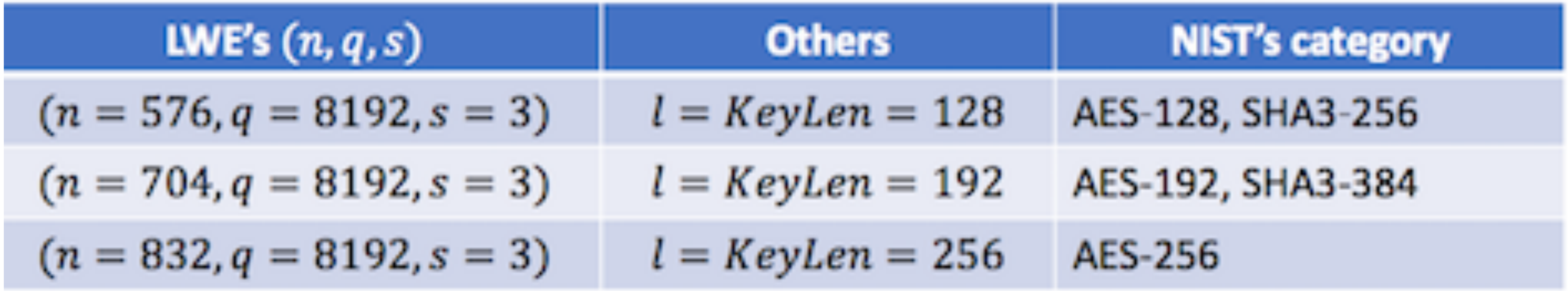
# Act I:

# How confident are we in our security claims?

THE DEVIL'S ADVOCATE

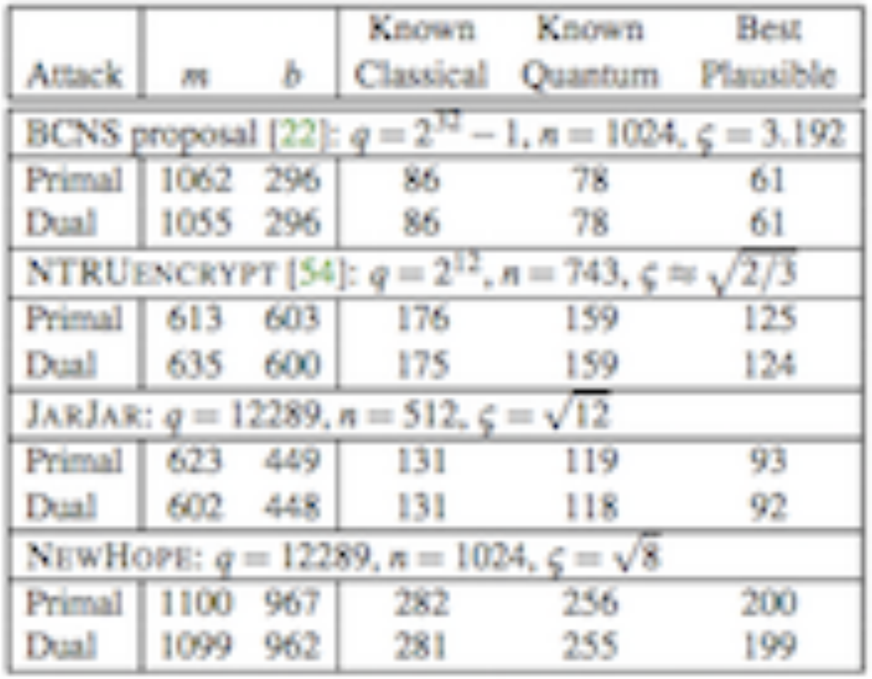
Adversary's
ADVOCATE

# Quantitative Security Claims

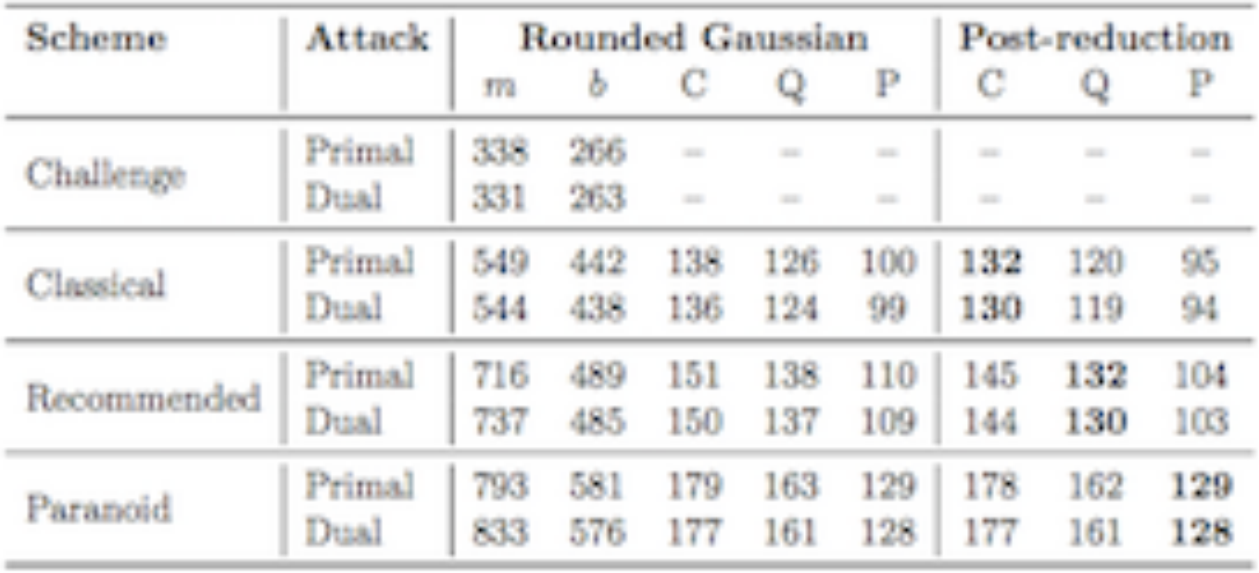


| LWE's $(n, q, s)$ | Others | NIST's category |
|---|---|---|
| $(n = 576, q = 8192, s = 3)$ | $l = KeyLen = 128$ | AES-128, SHA3-256 |
| $(n = 704, q = 8192, s = 3)$ | $l = KeyLen = 192$ | AES-192, SHA3-384 |
| $(n = 832, q = 8192, s = 3)$ | $l = KeyLen = 256$ | AES-256 |

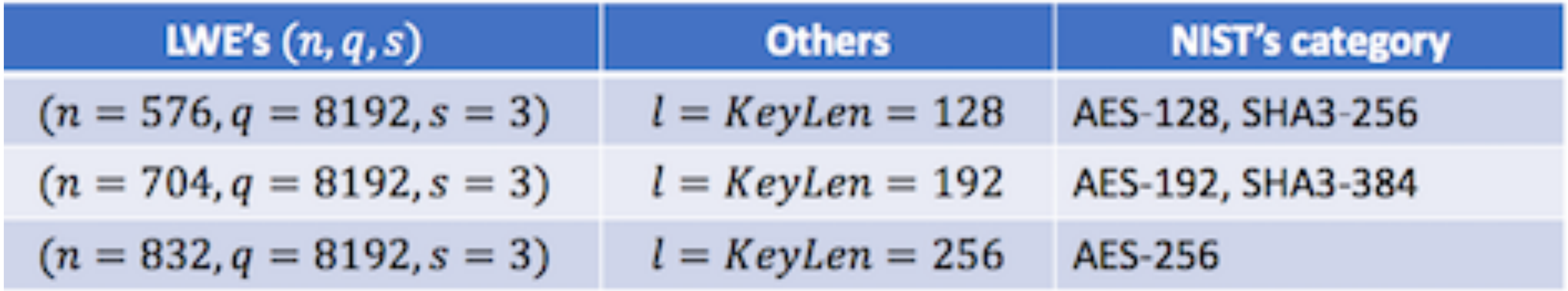
# Quantitative Security Claims



| Attack | $m$ | $b$ | Known Classical | Known Quantum | Best Plausible |
|---|---|---|---|---|---|
| BCNS proposal [22]: $q = 2^{32} - 1$, $n = 1024$, $\varsigma = 3.192$ | | | | | |
| Primal | 1062 | 296 | 86 | 78 | 61 |
| Dual | 1055 | 296 | 86 | 78 | 61 |
| NTRUENCRYPT [54]: $q = 2^{11}$, $n = 743$, $\varsigma \approx \sqrt{2/3}$ | | | | | |
| Primal | 613 | 603 | 176 | 159 | 125 |
| Dual | 635 | 600 | 175 | 159 | 124 |
| JARJAR: $q = 12289$, $n = 512$, $\varsigma = \sqrt{12}$ | | | | | |
| Primal | 623 | 449 | 131 | 119 | 93 |
| Dual | 602 | 448 | 131 | 118 | 92 |
| NEWHOPE: $q = 12289$, $n = 1024$, $\varsigma = \sqrt{8}$ | | | | | |
| Primal | 1100 | 967 | 282 | 256 | 200 |
| Dual | 1099 | 962 | 281 | 255 | 199 |



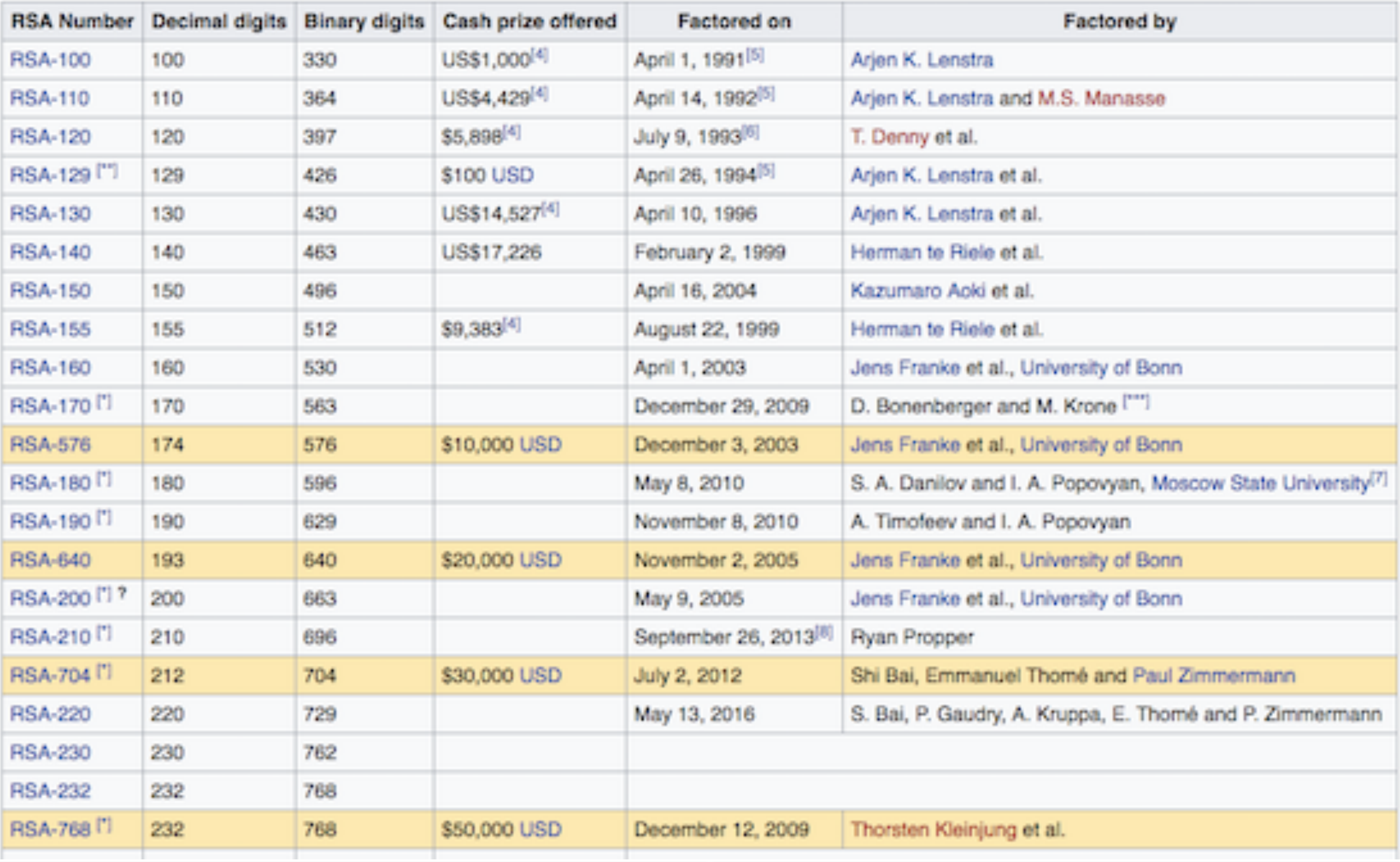
| Scheme | Attack | Rounded Gaussian | | | | | Post-reduction | | |
|---|---|---|---|---|---|---|---|---|---|
| | | $m$ | $b$ | C | Q | P | C | Q | P |
| Challenge | Primal | 338 | 266 | – | – | – | – | – | – |
| | Dual | 331 | 263 | – | – | – | – | – | – |
| Classical | Primal | 549 | 442 | 138 | 126 | 100 | **132** | 120 | 95 |
| | Dual | 544 | 438 | 136 | 124 | 99 | **130** | 119 | 94 |
| Recommended | Primal | 716 | 489 | 151 | 138 | 110 | 145 | **132** | 104 |
| | Dual | 737 | 485 | 150 | 137 | 109 | 144 | **130** | 103 |
| Paranoid | Primal | 793 | 581 | 179 | 163 | 129 | 178 | 162 | **129** |
| | Dual | 833 | 576 | 177 | 161 | 128 | 177 | 161 | **128** |



| LWE's $(n, q, s)$ | Others | NIST's category |
|---|---|---|
| $(n = 576, q = 8192, s = 3)$ | $l = KeyLen = 128$ | AES-128, SHA3-256 |
| $(n = 704, q = 8192, s = 3)$ | $l = KeyLen = 192$ | AES-192, SHA3-384 |
| $(n = 832, q = 8192, s = 3)$ | $l = KeyLen = 256$ | AES-256 |

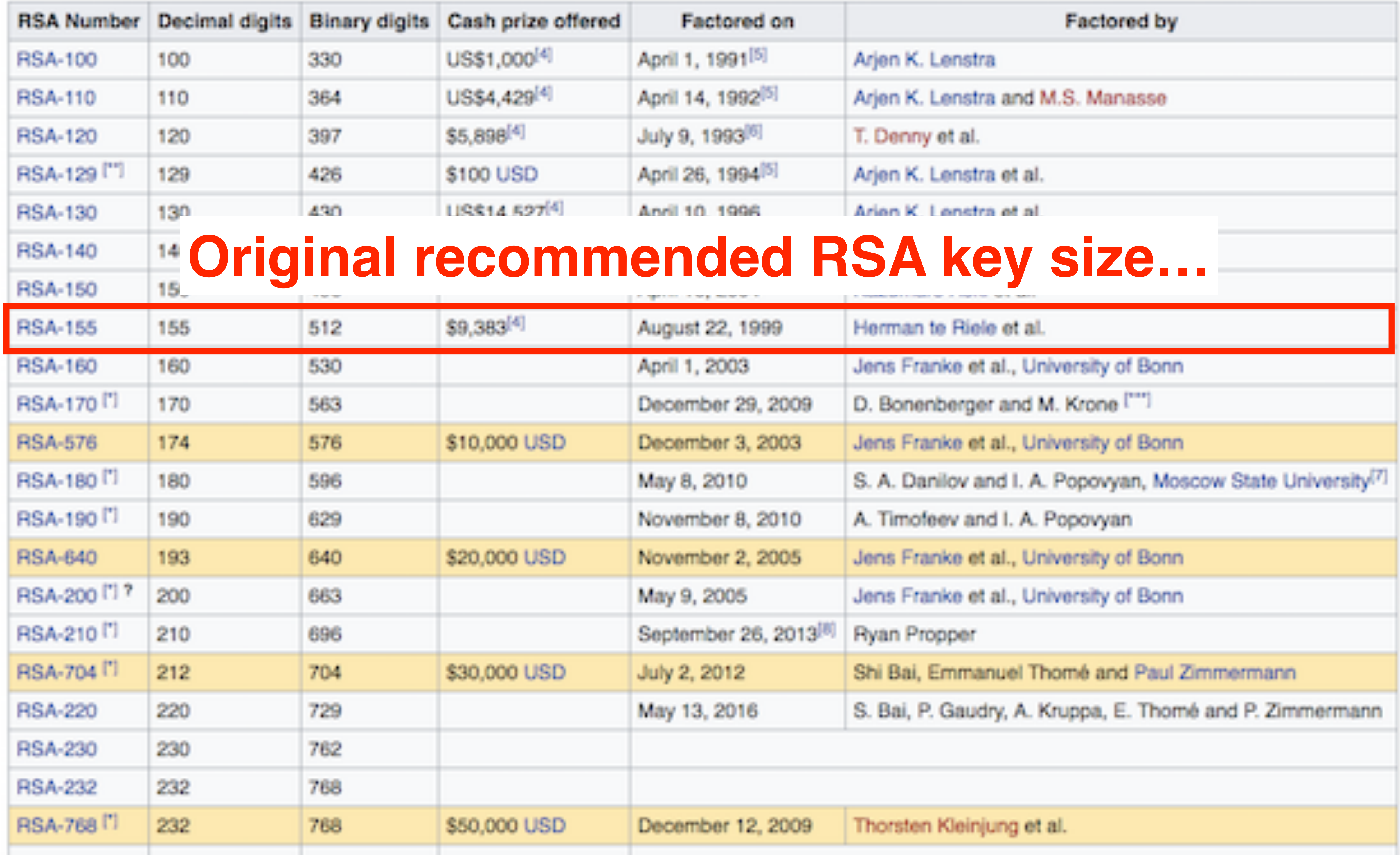
# Quantitative Security Is Hard…

# Quantitative Security Is Hard…



| RSA Number | Decimal digits | Binary digits | Cash prize offered | Factored on | Factored by |
|---|---|---|---|---|---|
| RSA-100 | 100 | 330 | US$1,000[4] | April 1, 1991[5] | Arjen K. Lenstra |
| RSA-110 | 110 | 364 | US$4,429[4] | April 14, 1992[5] | Arjen K. Lenstra and M.S. Manasse |
| RSA-120 | 120 | 397 | $5,898[4] | July 9, 1993[6] | T. Denny et al. |
| RSA-129 [**] | 129 | 426 | $100 USD | April 26, 1994[5] | Arjen K. Lenstra et al. |
| RSA-130 | 130 | 430 | US$14,527[4] | April 10, 1996 | Arjen K. Lenstra et al. |
| RSA-140 | 140 | 463 | US$17,226 | February 2, 1999 | Herman te Riele et al. |
| RSA-150 | 150 | 496 | | April 16, 2004 | Kazumaro Aoki et al. |
| RSA-155 | 155 | 512 | $9,383[4] | August 22, 1999 | Herman te Riele et al. |
| RSA-160 | 160 | 530 | | April 1, 2003 | Jens Franke et al., University of Bonn |
| RSA-170 [*] | 170 | 563 | | December 29, 2009 | D. Bonenberger and M. Krone [***] |
| RSA-576 | 174 | 576 | $10,000 USD | December 3, 2003 | Jens Franke et al., University of Bonn |
| RSA-180 [*] | 180 | 596 | | May 8, 2010 | S. A. Danilov and I. A. Popovyan, Moscow State University[7] |
| RSA-190 [*] | 190 | 629 | | November 8, 2010 | A. Timofeev and I. A. Popovyan |
| RSA-640 | 193 | 640 | $20,000 USD | November 2, 2005 | Jens Franke et al., University of Bonn |
| RSA-200 [*] ? | 200 | 663 | | May 9, 2005 | Jens Franke et al., University of Bonn |
| RSA-210 [*] | 210 | 696 | | September 26, 2013[8] | Ryan Propper |
| RSA-704 [*] | 212 | 704 | $30,000 USD | July 2, 2012 | Shi Bai, Emmanuel Thomé and Paul Zimmermann |
| RSA-220 | 220 | 729 | | May 13, 2016 | S. Bai, P. Gaudry, A. Kruppa, E. Thomé and P. Zimmermann |
| RSA-230 | 230 | 762 | | | |
| RSA-232 | 232 | 768 | | | |
| RSA-768 [*] | 232 | 768 | $50,000 USD | December 12, 2009 | Thorsten Kleinjung et al. |

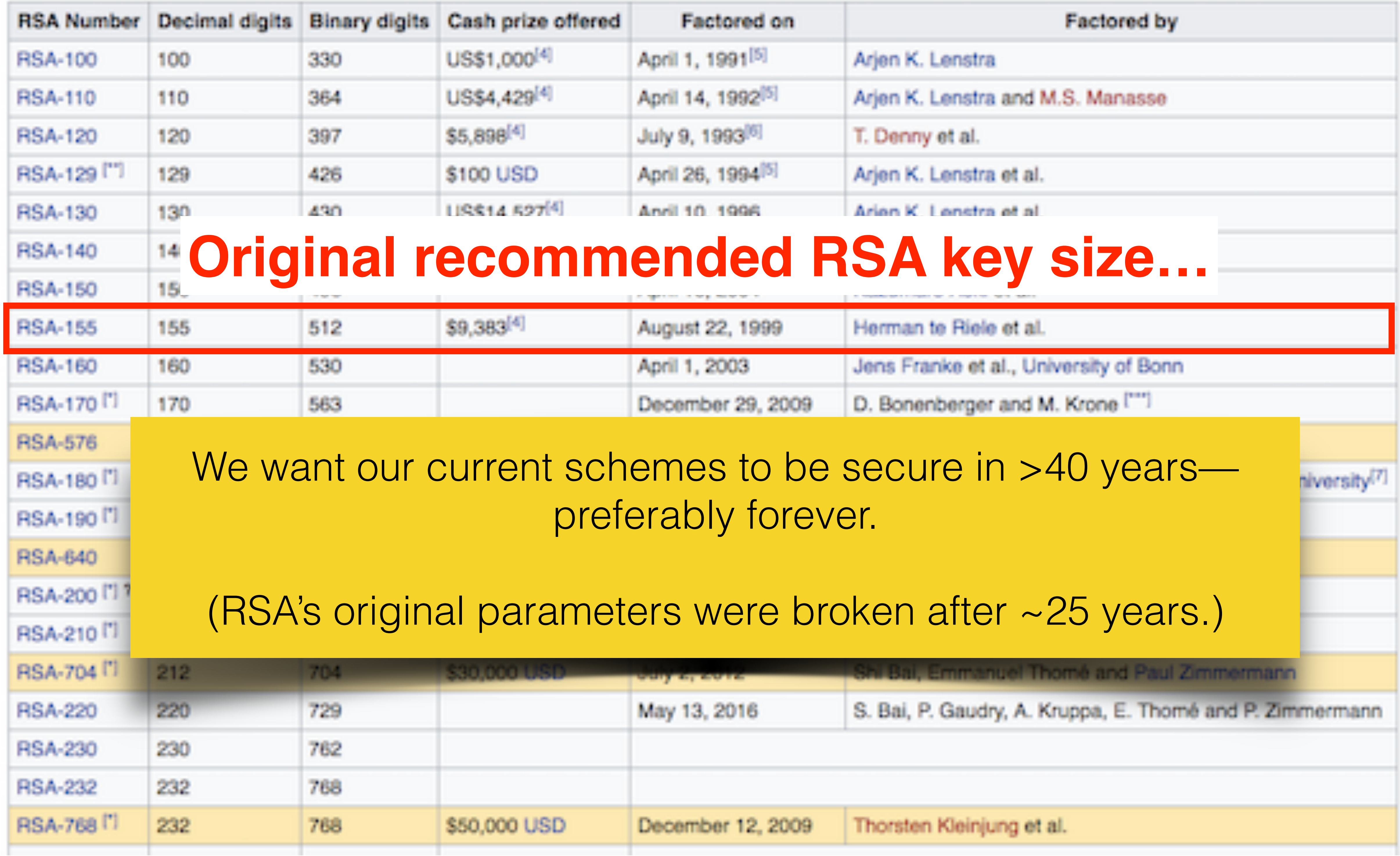
# Quantitative Security Is Hard…



| RSA Number | Decimal digits | Binary digits | Cash prize offered | Factored on | Factored by |
|---|---|---|---|---|---|
| RSA-100 | 100 | 330 | US$1,000[4] | April 1, 1991[5] | Arjen K. Lenstra |
| RSA-110 | 110 | 364 | US$4,429[4] | April 14, 1992[5] | Arjen K. Lenstra and M.S. Manasse |
| RSA-120 | 120 | 397 | $5,898[4] | July 9, 1993[6] | T. Denny et al. |
| RSA-129 [**] | 129 | 426 | $100 USD | April 26, 1994[5] | Arjen K. Lenstra et al. |
| RSA-130 | 130 | 430 | US$14,527[4] | April 10, 1996 | Arjen K. Lenstra et al. |
| RSA-140 | 14 | | | | |
| RSA-150 | 15 | | | | |
| RSA-155 | 155 | 512 | $9,383[4] | August 22, 1999 | Herman te Riele et al. |
| RSA-160 | 160 | 530 | | April 1, 2003 | Jens Franke et al., University of Bonn |
| RSA-170 [*] | 170 | 563 | | December 29, 2009 | D. Bonenberger and M. Krone [***] |
| RSA-576 | 174 | 576 | $10,000 USD | December 3, 2003 | Jens Franke et al., University of Bonn |
| RSA-180 [*] | 180 | 596 | | May 8, 2010 | S. A. Danilov and I. A. Popovyan, Moscow State University[7] |
| RSA-190 [*] | 190 | 629 | | November 8, 2010 | A. Timofeev and I. A. Popovyan |
| RSA-640 | 193 | 640 | $20,000 USD | November 2, 2005 | Jens Franke et al., University of Bonn |
| RSA-200 [*] ? | 200 | 663 | | May 9, 2005 | Jens Franke et al., University of Bonn |
| RSA-210 [*] | 210 | 696 | | September 26, 2013[8] | Ryan Propper |
| RSA-704 [*] | 212 | 704 | $30,000 USD | July 2, 2012 | Shi Bai, Emmanuel Thomé and Paul Zimmermann |
| RSA-220 | 220 | 729 | | May 13, 2016 | S. Bai, P. Gaudry, A. Kruppa, E. Thomé and P. Zimmermann |
| RSA-230 | 230 | 762 | | | |
| RSA-232 | 232 | 768 | | | |
| RSA-768 [*] | 232 | 768 | $50,000 USD | December 12, 2009 | Thorsten Kleinjung et al. |

**Original recommended RSA key size…**

# Quantitative Security Is Hard…



| RSA Number | Decimal digits | Binary digits | Cash prize offered | Factored on | Factored by |
|---|---|---|---|---|---|
| RSA-100 | 100 | 330 | US$1,000[4] | April 1, 1991[5] | Arjen K. Lenstra |
| RSA-110 | 110 | 364 | US$4,429[4] | April 14, 1992[5] | Arjen K. Lenstra and M.S. Manasse |
| RSA-120 | 120 | 397 | $5,898[4] | July 9, 1993[6] | T. Denny et al. |
| RSA-129 [**] | 129 | 426 | $100 USD | April 26, 1994[5] | Arjen K. Lenstra et al. |
| RSA-130 | 130 | 430 | US$14,527[4] | April 10, 1996 | Arjen K. Lenstra et al. |
| RSA-140 | 14 | | | | |
| RSA-150 | 15 | | | | |
| RSA-155 | 155 | 512 | $9,383[4] | August 22, 1999 | Herman te Riele et al. |
| RSA-160 | 160 | 530 | | April 1, 2003 | Jens Franke et al., University of Bonn |
| RSA-170 [*] | 170 | 563 | | December 29, 2009 | D. Bonenberger and M. Krone [***] |
| RSA-576 | | | | | |
| RSA-180 [*] | | | | | niversity[7] |
| RSA-190 [*] | | | | | |
| RSA-640 | | | | | |
| RSA-200 [*] | | | | | |
| RSA-210 [*] | | | | | |
| RSA-704 [*] | 212 | 704 | $30,000 USD | July 2, 2012 | Shi Bai, Emmanuel Thomé and Paul Zimmermann |
| RSA-220 | 220 | 729 | | May 13, 2016 | S. Bai, P. Gaudry, A. Kruppa, E. Thomé and P. Zimmermann |
| RSA-230 | 230 | 762 | | | |
| RSA-232 | 232 | 768 | | | |
| RSA-768 [*] | 232 | 768 | $50,000 USD | December 12, 2009 | Thorsten Kleinjung et al. |

**Original recommended RSA key size…**

We want our current schemes to be secure in >40 years—preferably forever.

(RSA's original parameters were broken after ~25 years.)

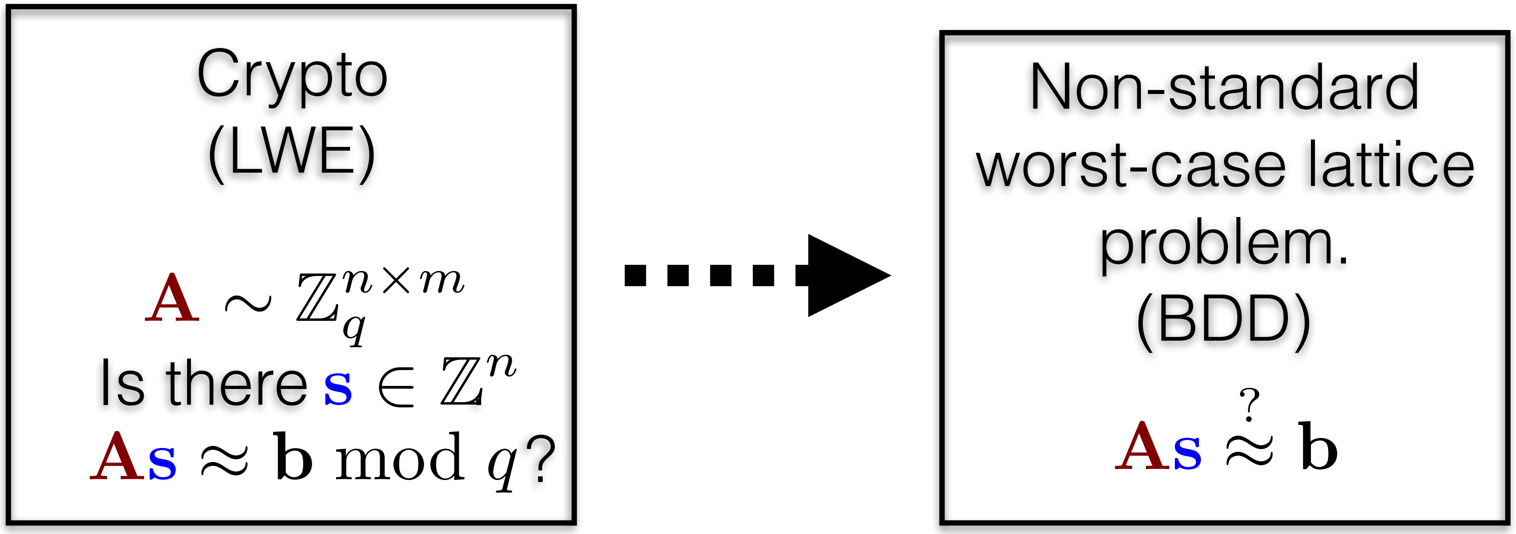
# Security of Lattice-Based Crypto (a caricature)

# Security of Lattice-Based Crypto (a caricature)

Crypto
(LWE)

$\mathbf{A} \sim \mathbb{Z}_q^{n \times m}$
Is there $\mathbf{s} \in \mathbb{Z}^n$
$\mathbf{A}\mathbf{s} \approx \mathbf{b} \bmod q$?

# Security of Lattice-Based Crypto (a caricature)



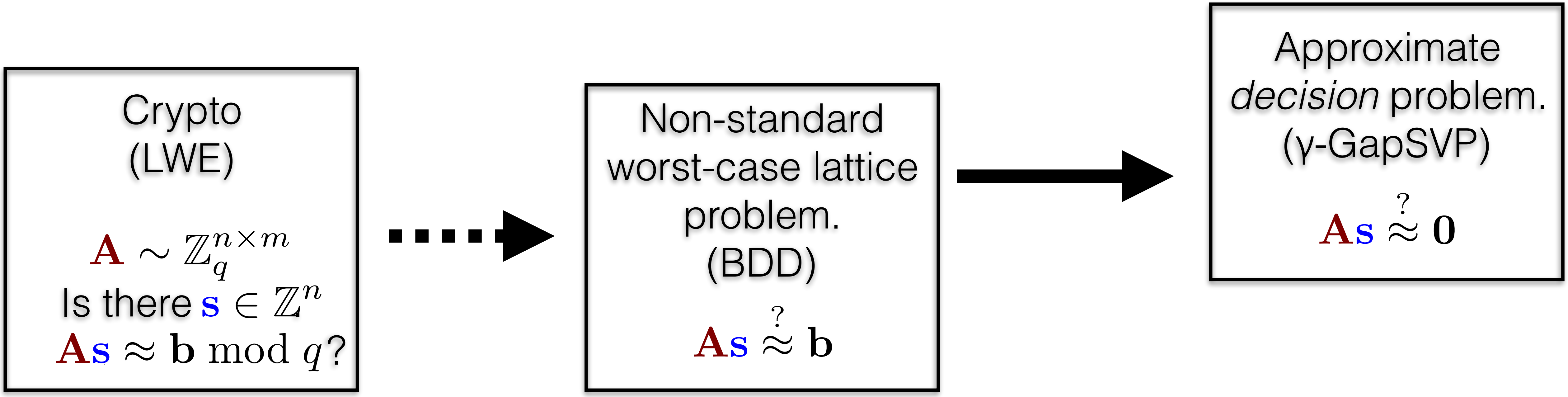
Crypto
(LWE)

$\mathbf{A} \sim \mathbb{Z}_q^{n \times m}$
Is there $\mathbf{s} \in \mathbb{Z}^n$
$\mathbf{A}\mathbf{s} \approx \mathbf{b} \bmod q$?

Non-standard worst-case lattice problem.
(BDD)

$\mathbf{A}\mathbf{s} \overset{?}{\approx} \mathbf{b}$

# Security of Lattice-Based Crypto (a caricature)



Crypto (LWE)

$\mathbf{A} \sim \mathbb{Z}_q^{n\times m}$

Is there $\mathbf{s} \in \mathbb{Z}^n$

$\mathbf{A}\mathbf{s} \approx \mathbf{b} \bmod q$?

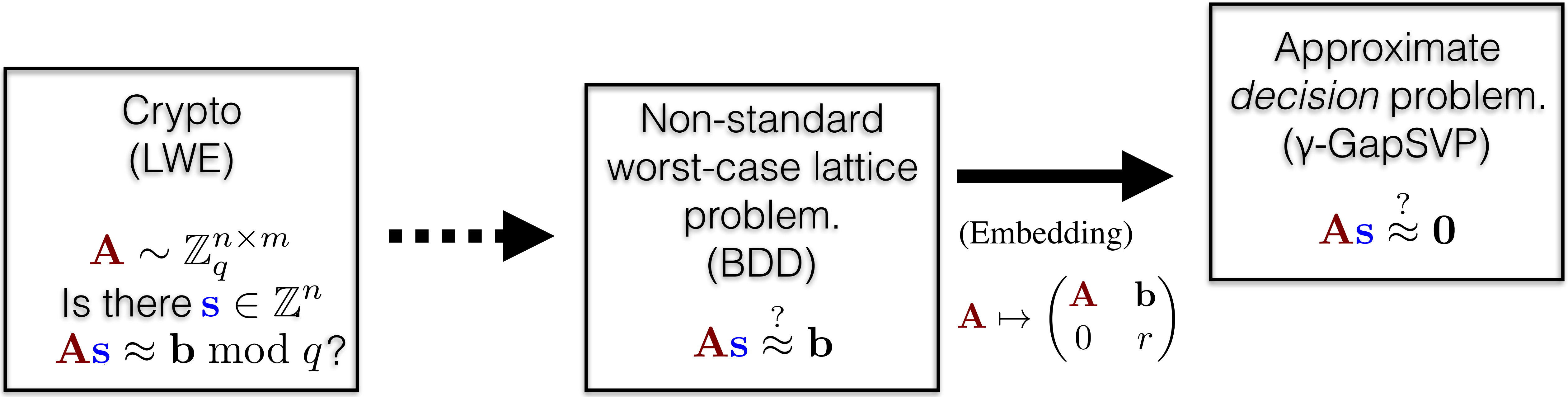
Non-standard worst-case lattice problem. (BDD)

$\mathbf{A}\mathbf{s} \overset{?}{\approx} \mathbf{b}$

Approximate *decision* problem. (γ-GapSVP)

$\mathbf{A}\mathbf{s} \overset{?}{\approx} \mathbf{0}$

# Security of Lattice-Based Crypto (a caricature)



Crypto (LWE)

$\mathbf{A} \sim \mathbb{Z}_q^{n\times m}$

Is there $\mathbf{s} \in \mathbb{Z}^n$

$\mathbf{A}\mathbf{s} \approx \mathbf{b} \bmod q$?

Non-standard worst-case lattice problem. (BDD)

$\mathbf{A}\mathbf{s} \overset{?}{\approx} \mathbf{b}$

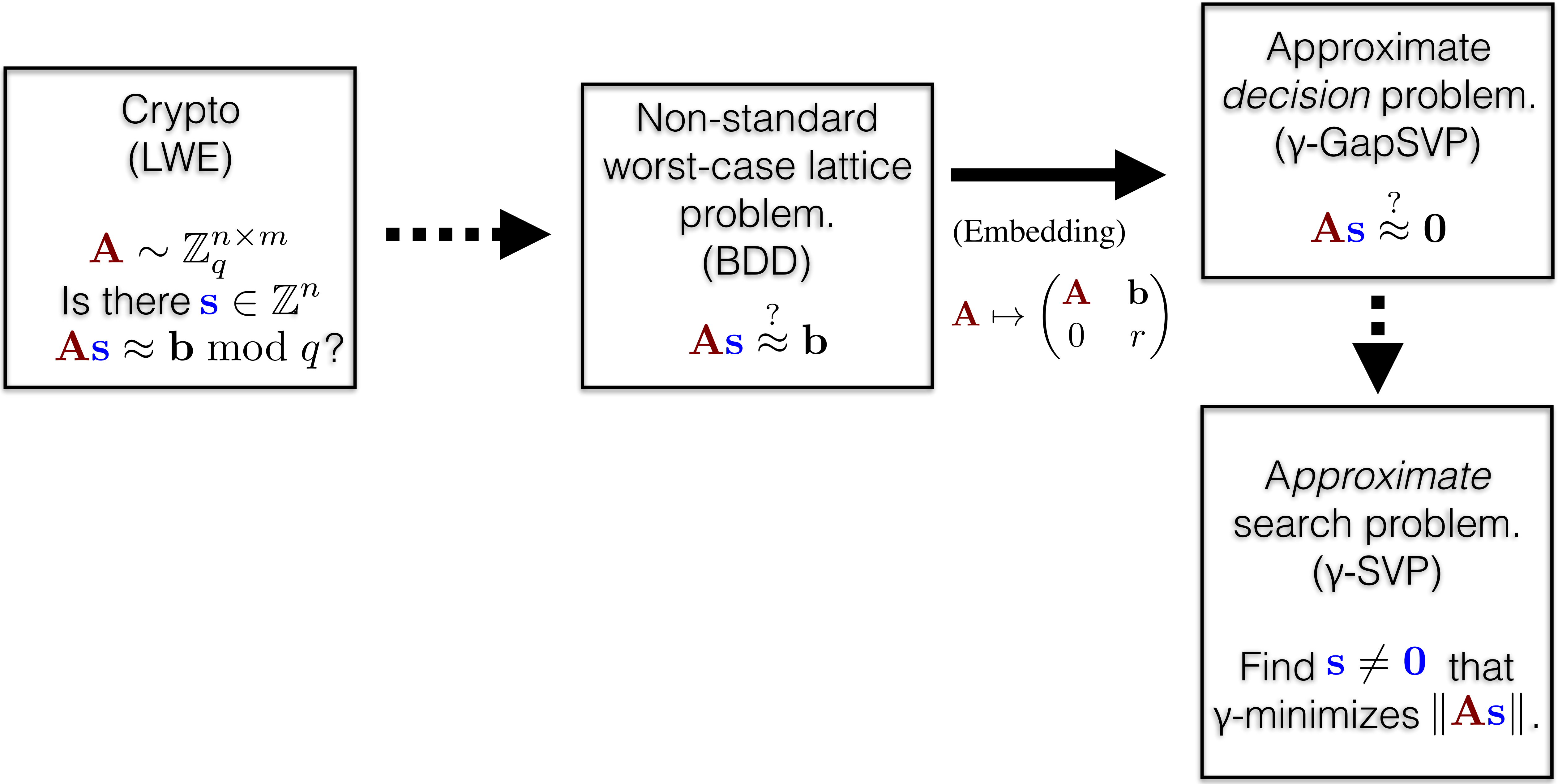
(Embedding)

$\mathbf{A} \mapsto \begin{pmatrix} \mathbf{A} & \mathbf{b} \\ 0 & r \end{pmatrix}$

Approximate *decision* problem. (γ-GapSVP)

$\mathbf{A}\mathbf{s} \overset{?}{\approx} \mathbf{0}$

# Security of Lattice-Based Crypto (a caricature)



Crypto (LWE)

$\mathbf{A} \sim \mathbb{Z}_q^{n \times m}$
Is there $\mathbf{s} \in \mathbb{Z}^n$
$\mathbf{A}\mathbf{s} \approx \mathbf{b} \bmod q$?

Non-standard worst-case lattice problem. (BDD)

$\mathbf{A}\mathbf{s} \overset{?}{\approx} \mathbf{b}$

(Embedding)

$\mathbf{A} \mapsto \begin{pmatrix} \mathbf{A} & \mathbf{b} \\ 0 & r \end{pmatrix}$

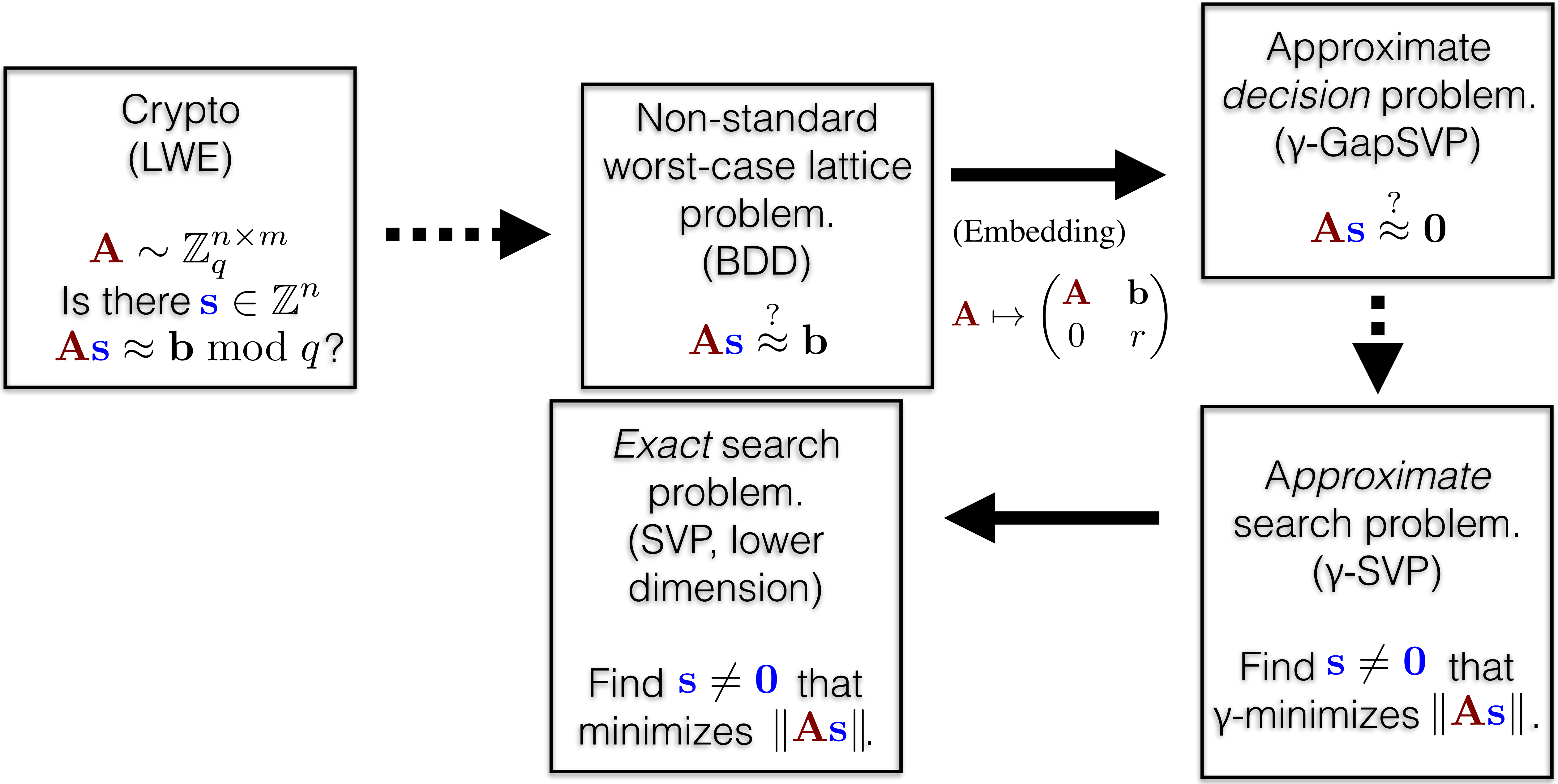
Approximate *decision* problem. (γ-GapSVP)

$\mathbf{A}\mathbf{s} \overset{?}{\approx} \mathbf{0}$

*Approximate* search problem. (γ-SVP)

Find $\mathbf{s} \neq \mathbf{0}$ that γ-minimizes $\|\mathbf{A}\mathbf{s}\|$.

# Security of Lattice-Based Crypto (a caricature)



Crypto (LWE)

$\mathbf{A} \sim \mathbb{Z}_q^{n \times m}$

Is there $\mathbf{s} \in \mathbb{Z}^n$

$\mathbf{A}\mathbf{s} \approx \mathbf{b} \bmod q$?

Non-standard worst-case lattice problem. (BDD)

$\mathbf{A}\mathbf{s} \overset{?}{\approx} \mathbf{b}$

(Embedding)

$\mathbf{A} \mapsto \begin{pmatrix} \mathbf{A} & \mathbf{b} \\ 0 & r \end{pmatrix}$

Approximate *decision* problem. (γ-GapSVP)

$\mathbf{A}\mathbf{s} \overset{?}{\approx} \mathbf{0}$

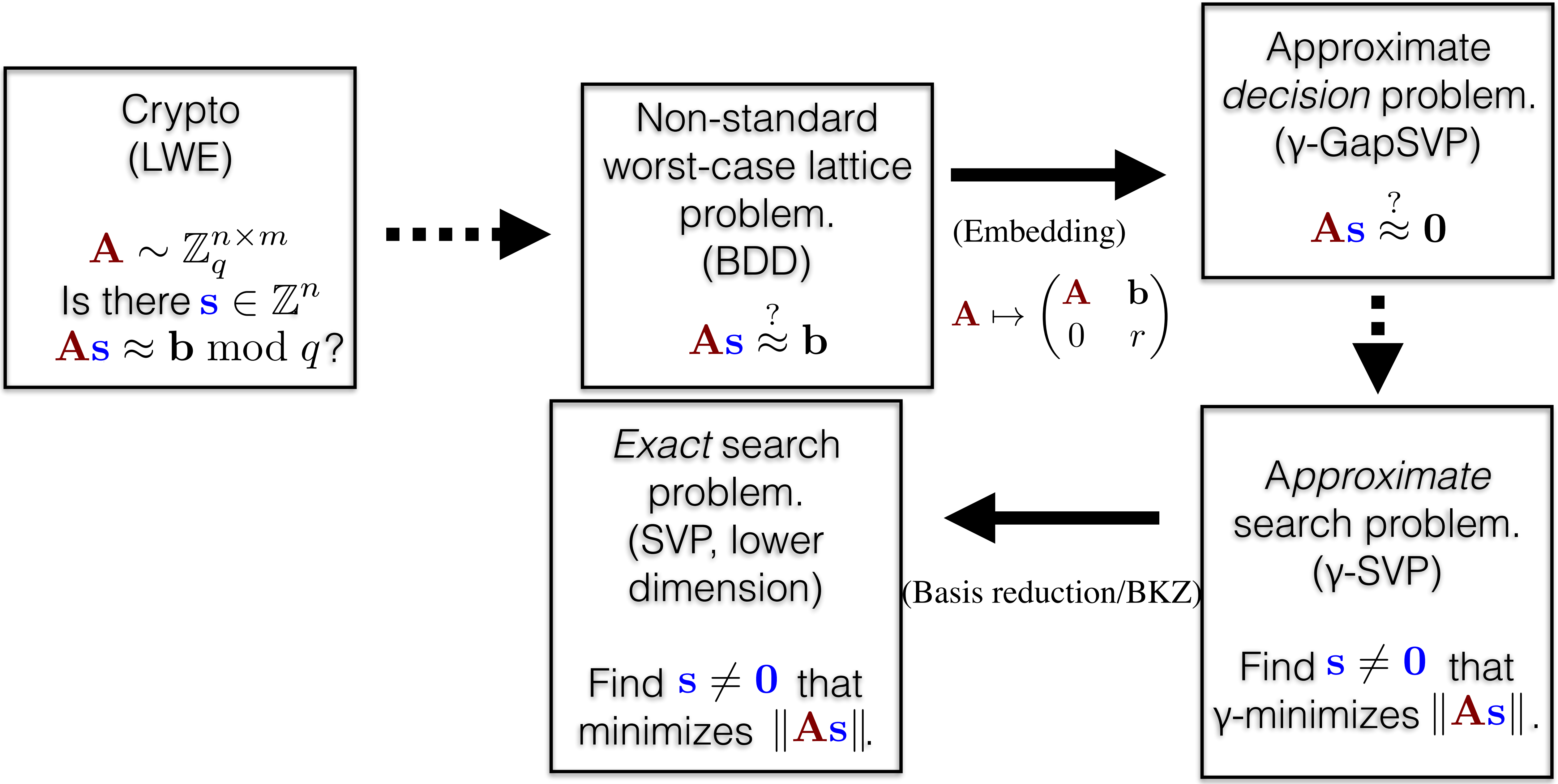
*Approximate* search problem. (γ-SVP)

Find $\mathbf{s} \neq \mathbf{0}$ that γ-minimizes $\|\mathbf{A}\mathbf{s}\|$.

*Exact* search problem. (SVP, lower dimension)

Find $\mathbf{s} \neq \mathbf{0}$ that minimizes $\|\mathbf{A}\mathbf{s}\|$.

# Security of Lattice-Based Crypto (a caricature)



Crypto (LWE)

$\mathbf{A} \sim \mathbb{Z}_q^{n \times m}$

Is there $\mathbf{s} \in \mathbb{Z}^n$

$\mathbf{A}\mathbf{s} \approx \mathbf{b} \bmod q$?

Non-standard worst-case lattice problem. (BDD)

$\mathbf{A}\mathbf{s} \overset{?}{\approx} \mathbf{b}$

(Embedding)

$\mathbf{A} \mapsto \begin{pmatrix} \mathbf{A} & \mathbf{b} \\ 0 & r \end{pmatrix}$

Approximate *decision* problem. (γ-GapSVP)

$\mathbf{A}\mathbf{s} \overset{?}{\approx} \mathbf{0}$

*Approximate* search problem. (γ-SVP)

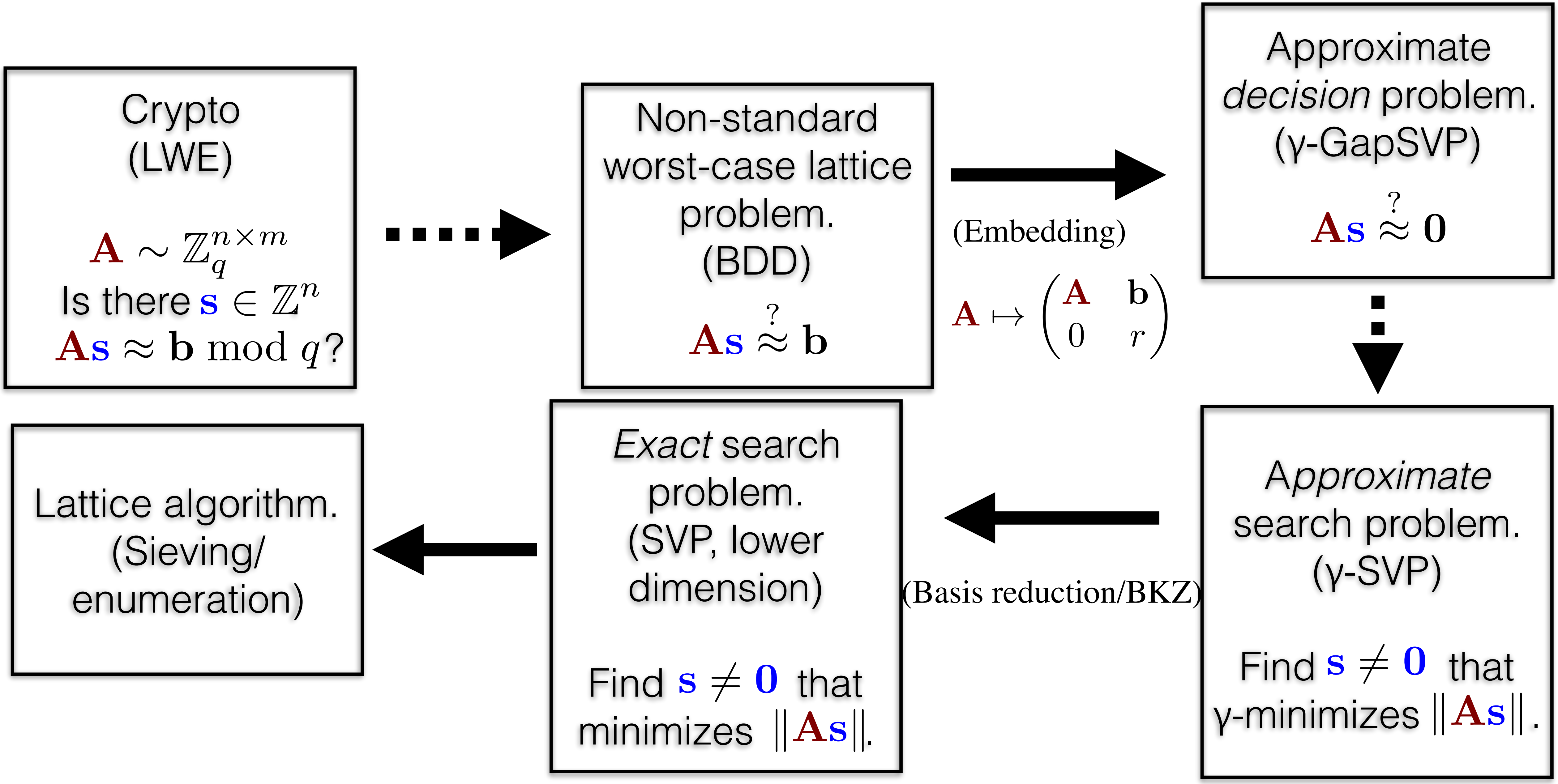
Find $\mathbf{s} \neq \mathbf{0}$ that γ-minimizes $\|\mathbf{A}\mathbf{s}\|$.

(Basis reduction/BKZ)

*Exact* search problem. (SVP, lower dimension)

Find $\mathbf{s} \neq \mathbf{0}$ that minimizes $\|\mathbf{A}\mathbf{s}\|$.

# Security of Lattice-Based Crypto (a caricature)



Crypto (LWE)

$\mathbf{A} \sim \mathbb{Z}_q^{n \times m}$

Is there $\mathbf{s} \in \mathbb{Z}^n$

$\mathbf{A}\mathbf{s} \approx \mathbf{b} \bmod q$?

Non-standard worst-case lattice problem. (BDD)

$\mathbf{A}\mathbf{s} \overset{?}{\approx} \mathbf{b}$

(Embedding)

$\mathbf{A} \mapsto \begin{pmatrix} \mathbf{A} & \mathbf{b} \\ 0 & r \end{pmatrix}$

Approximate *decision* problem. (γ-GapSVP)

$\mathbf{A}\mathbf{s} \overset{?}{\approx} \mathbf{0}$

*Approximate* search problem. (γ-SVP)

Find $\mathbf{s} \neq \mathbf{0}$ that γ-minimizes $\|\mathbf{A}\mathbf{s}\|$.

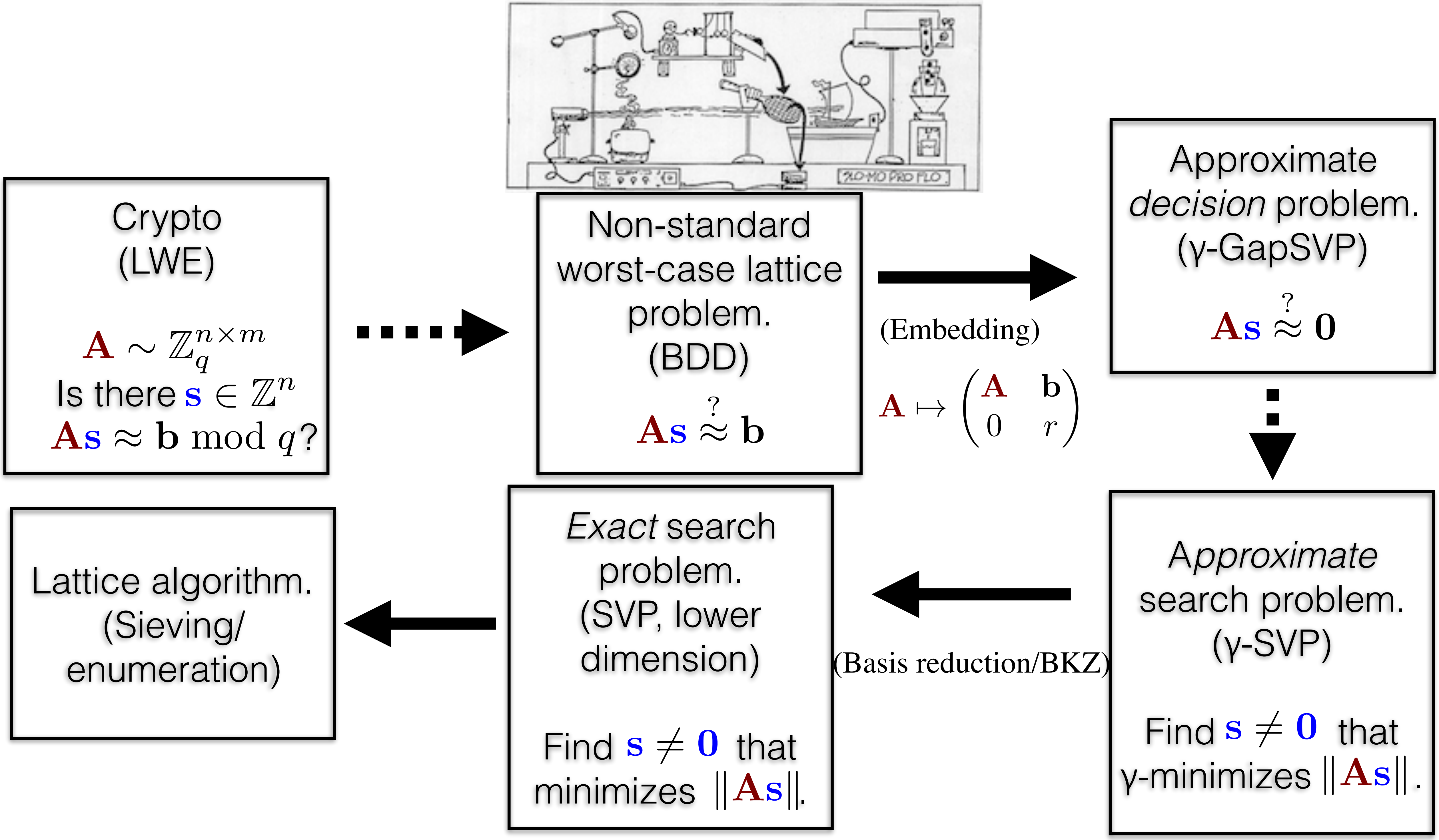
(Basis reduction/BKZ)

*Exact* search problem. (SVP, lower dimension)

Find $\mathbf{s} \neq \mathbf{0}$ that minimizes $\|\mathbf{A}\mathbf{s}\|$.

Lattice algorithm. (Sieving/ enumeration)

# Security of Lattice-Based Crypto (a caricature)



Crypto (LWE)

$\mathbf{A} \sim \mathbb{Z}_q^{n\times m}$

Is there $\mathbf{s} \in \mathbb{Z}^n$

$\mathbf{A}\mathbf{s} \approx \mathbf{b} \bmod q$?

Non-standard worst-case lattice problem. (BDD)

$\mathbf{A}\mathbf{s} \overset{?}{\approx} \mathbf{b}$

(Embedding)

$\mathbf{A} \mapsto \begin{pmatrix} \mathbf{A} & \mathbf{b} \\ 0 & r \end{pmatrix}$

Approximate *decision* problem. (γ-GapSVP)

$\mathbf{A}\mathbf{s} \overset{?}{\approx} \mathbf{0}$

*Approximate* search problem. (γ-SVP)

Find $\mathbf{s} \neq \mathbf{0}$ that γ-minimizes $\|\mathbf{A}\mathbf{s}\|$.

(Basis reduction/BKZ)

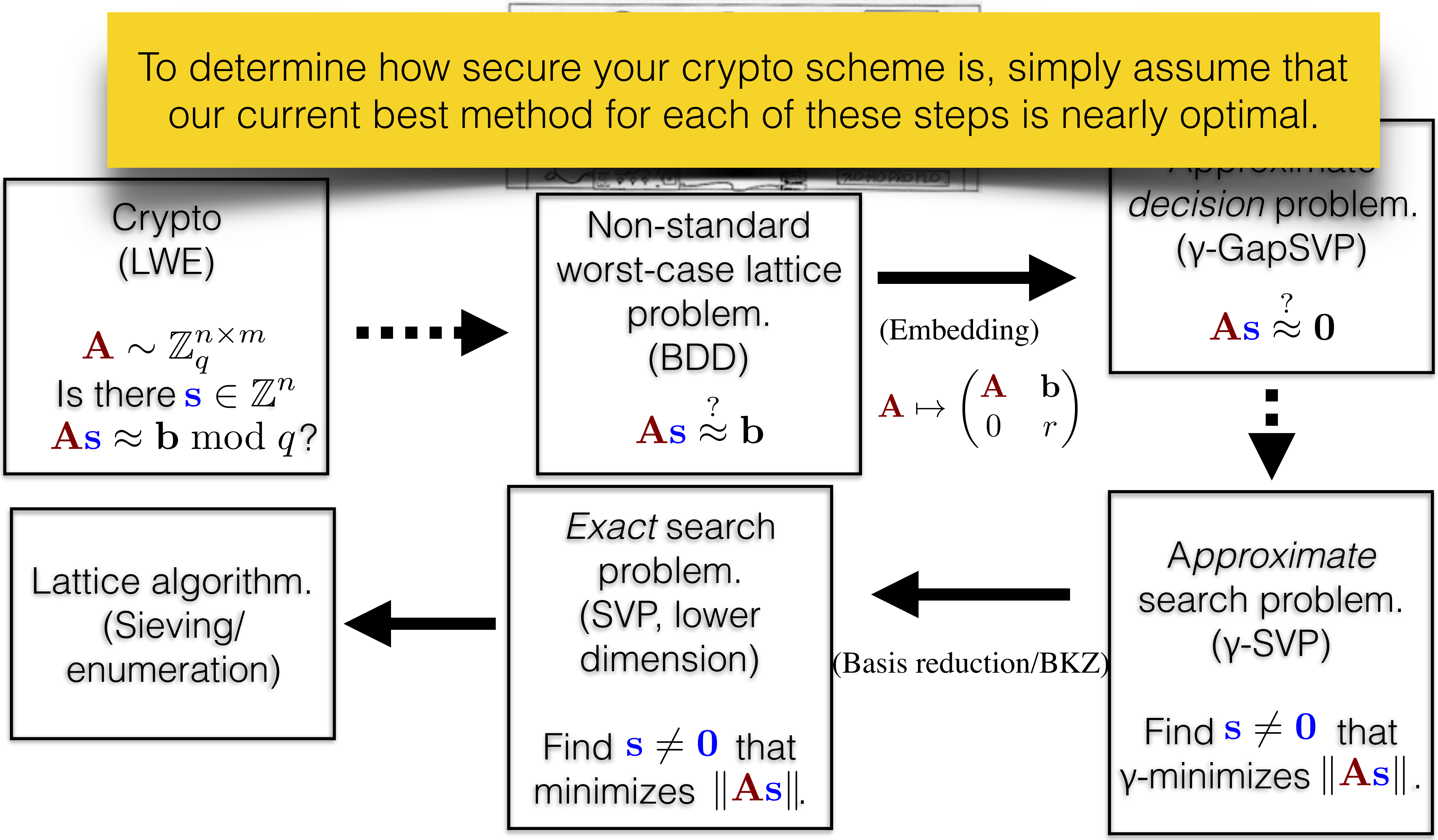
*Exact* search problem. (SVP, lower dimension)

Find $\mathbf{s} \neq \mathbf{0}$ that minimizes $\|\mathbf{A}\mathbf{s}\|$.

Lattice algorithm. (Sieving/ enumeration)

# Security of Lattice-Based Crypto (a caricature)

To determine how secure your crypto scheme is, simply assume that our current best method for each of these steps is nearly optimal.



Crypto (LWE)

$\mathbf{A} \sim \mathbb{Z}_q^{n \times m}$

Is there $\mathbf{s} \in \mathbb{Z}^n$

$\mathbf{A}\mathbf{s} \approx \mathbf{b} \bmod q$?

Non-standard worst-case lattice problem. (BDD)

$\mathbf{A}\mathbf{s} \overset{?}{\approx} \mathbf{b}$

(Embedding)

$\mathbf{A} \mapsto \begin{pmatrix} \mathbf{A} & \mathbf{b} \\ 0 & r \end{pmatrix}$

*Approximate* *decision* problem. (γ-GapSVP)

$\mathbf{A}\mathbf{s} \overset{?}{\approx} \mathbf{0}$

*Approximate* search problem. (γ-SVP)

Find $\mathbf{s} \neq \mathbf{0}$ that γ-minimizes $\|\mathbf{A}\mathbf{s}\|$.

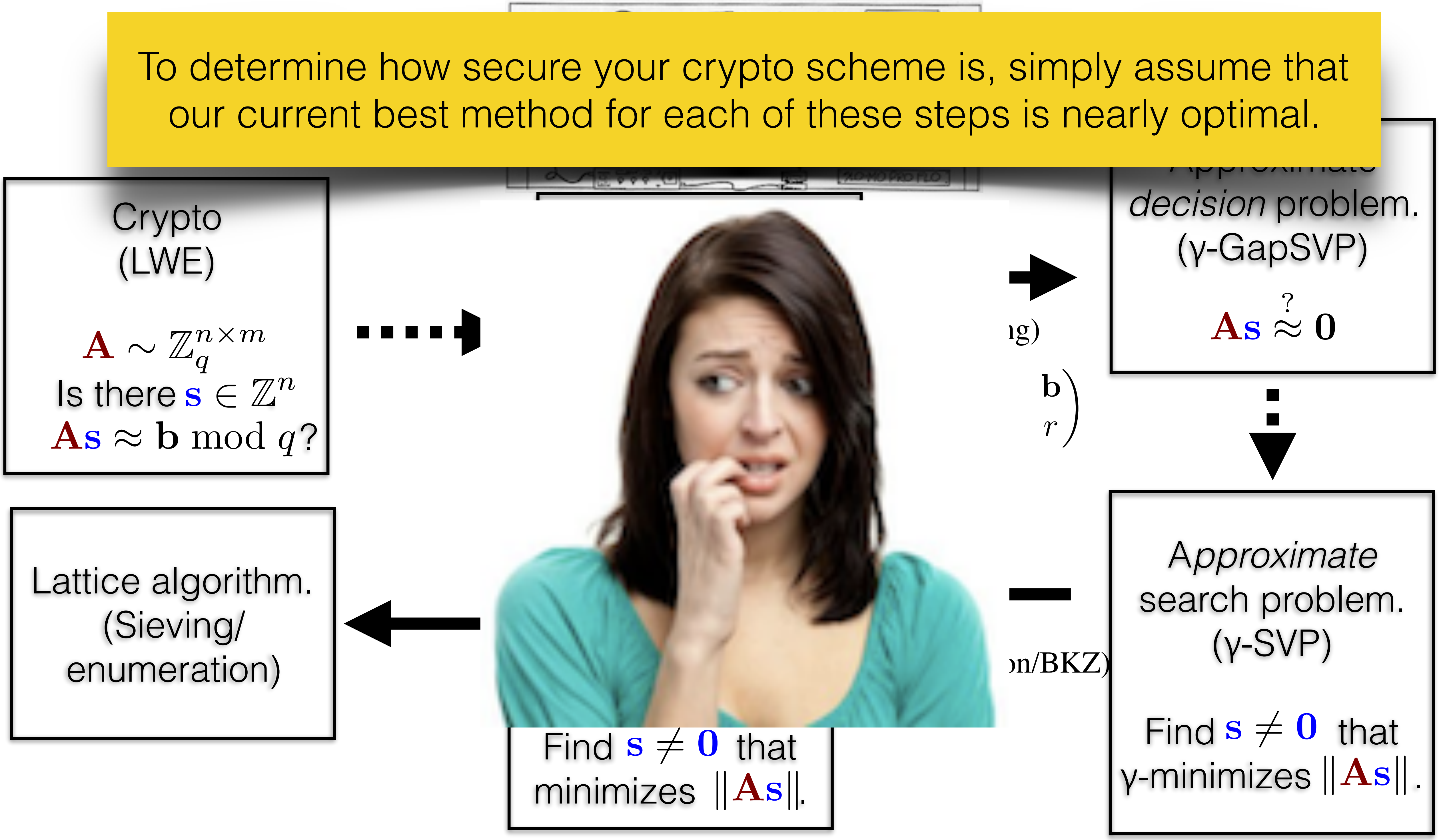
(Basis reduction/BKZ)

*Exact* search problem. (SVP, lower dimension)

Find $\mathbf{s} \neq \mathbf{0}$ that minimizes $\|\mathbf{A}\mathbf{s}\|$.

Lattice algorithm. (Sieving/ enumeration)

# Security of Lattice-Based Crypto (a caricature)



To determine how secure your crypto scheme is, simply assume that our current best method for each of these steps is nearly optimal.

Crypto (LWE)

$\mathbf{A} \sim \mathbb{Z}_q^{n \times m}$

Is there $\mathbf{s} \in \mathbb{Z}^n$

$\mathbf{A}\mathbf{s} \approx \mathbf{b} \bmod q$?

ıg)

$\mathbf{b}$

$r$

Approximate *decision* problem. (γ-GapSVP)

$\mathbf{A}\mathbf{s} \overset{?}{\approx} \mathbf{0}$

*Approximate* search problem. (γ-SVP)

Find $\mathbf{s} \neq \mathbf{0}$ that γ-minimizes $\|\mathbf{A}\mathbf{s}\|$.

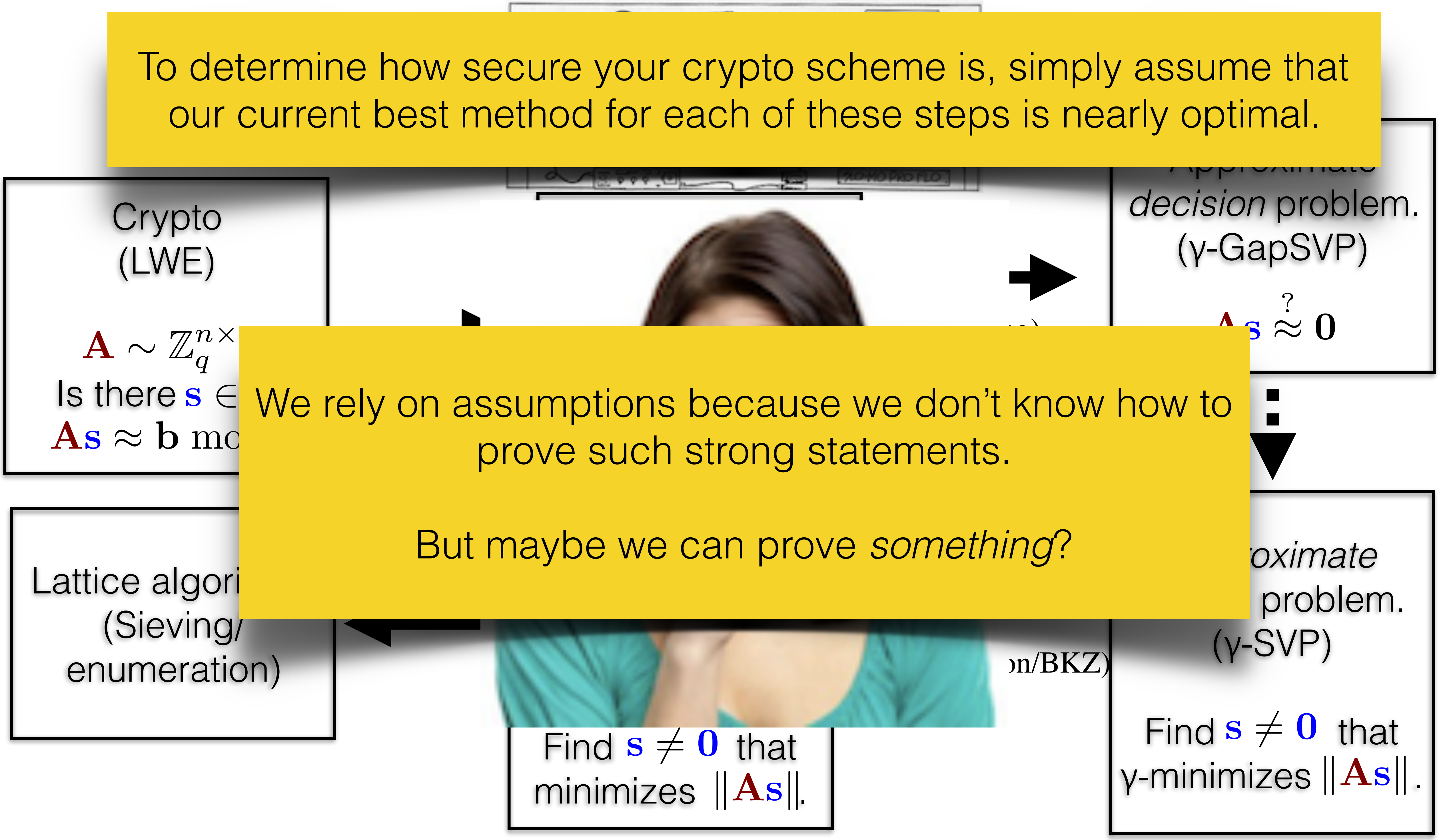
ɔn/BKZ)

Find $\mathbf{s} \neq \mathbf{0}$ that minimizes $\|\mathbf{A}\mathbf{s}\|$.

Lattice algorithm. (Sieving/ enumeration)

# Security of Lattice-Based Crypto (a caricature)

To determine how secure your crypto scheme is, simply assume that our current best method for each of these steps is nearly optimal.



Crypto (LWE)

$\mathbf{A} \sim \mathbb{Z}_q^{n\times}$

Is there $\mathbf{s} \in$

$\mathbf{A}\mathbf{s} \approx \mathbf{b}$ mo

Approximate *decision* problem. (γ-GapSVP)

$\mathbf{A}\mathbf{s} \overset{?}{\approx} \mathbf{0}$

*oximate* problem. (γ-SVP)

Find $\mathbf{s} \neq \mathbf{0}$ that γ-minimizes $\|\mathbf{A}\mathbf{s}\|$.

n/BKZ)

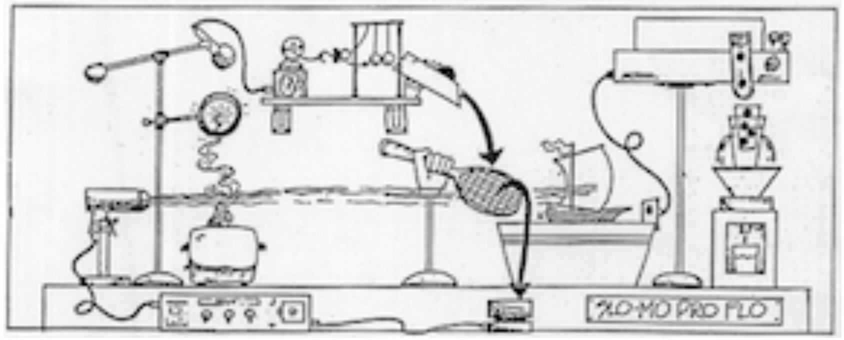
Find $\mathbf{s} \neq \mathbf{0}$ that minimizes $\|\mathbf{A}\mathbf{s}\|$.

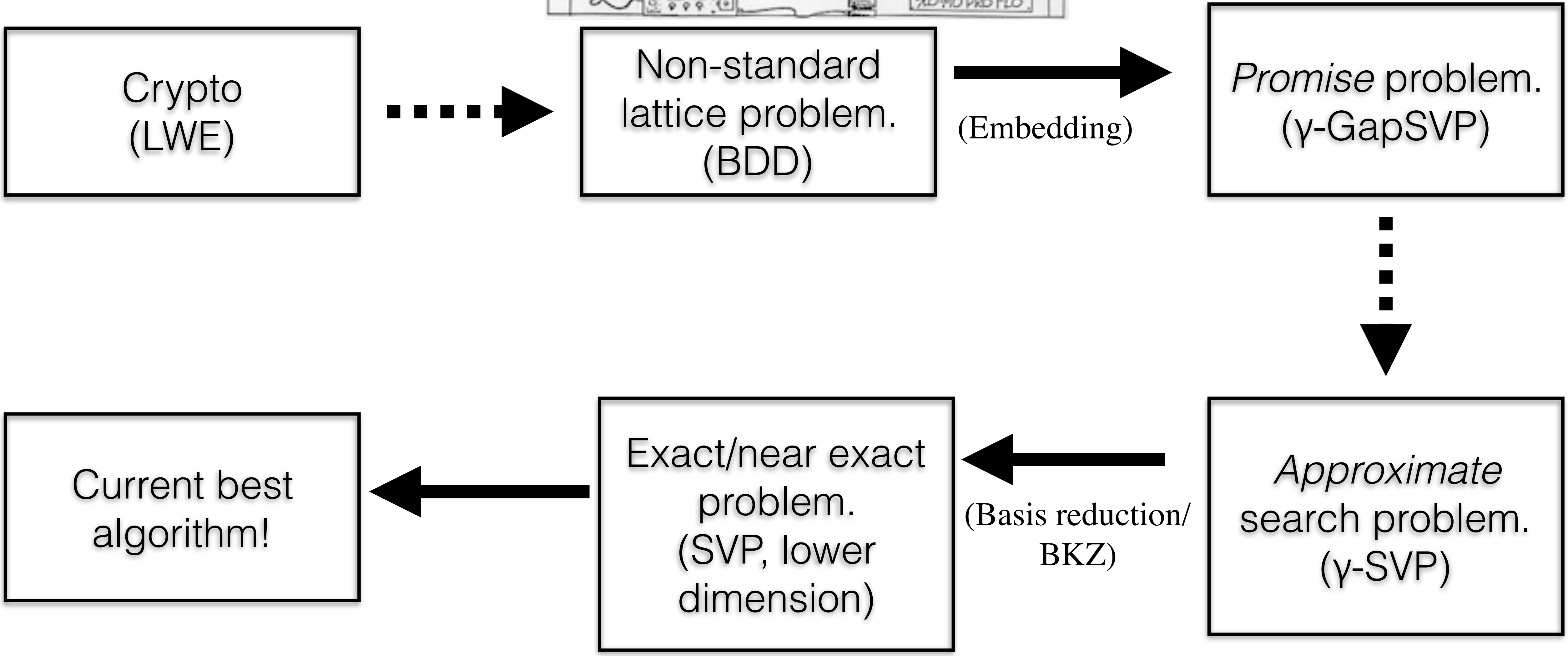
Lattice algori (Sieving/ enumeration)

We rely on assumptions because we don’t know how to prove such strong statements.

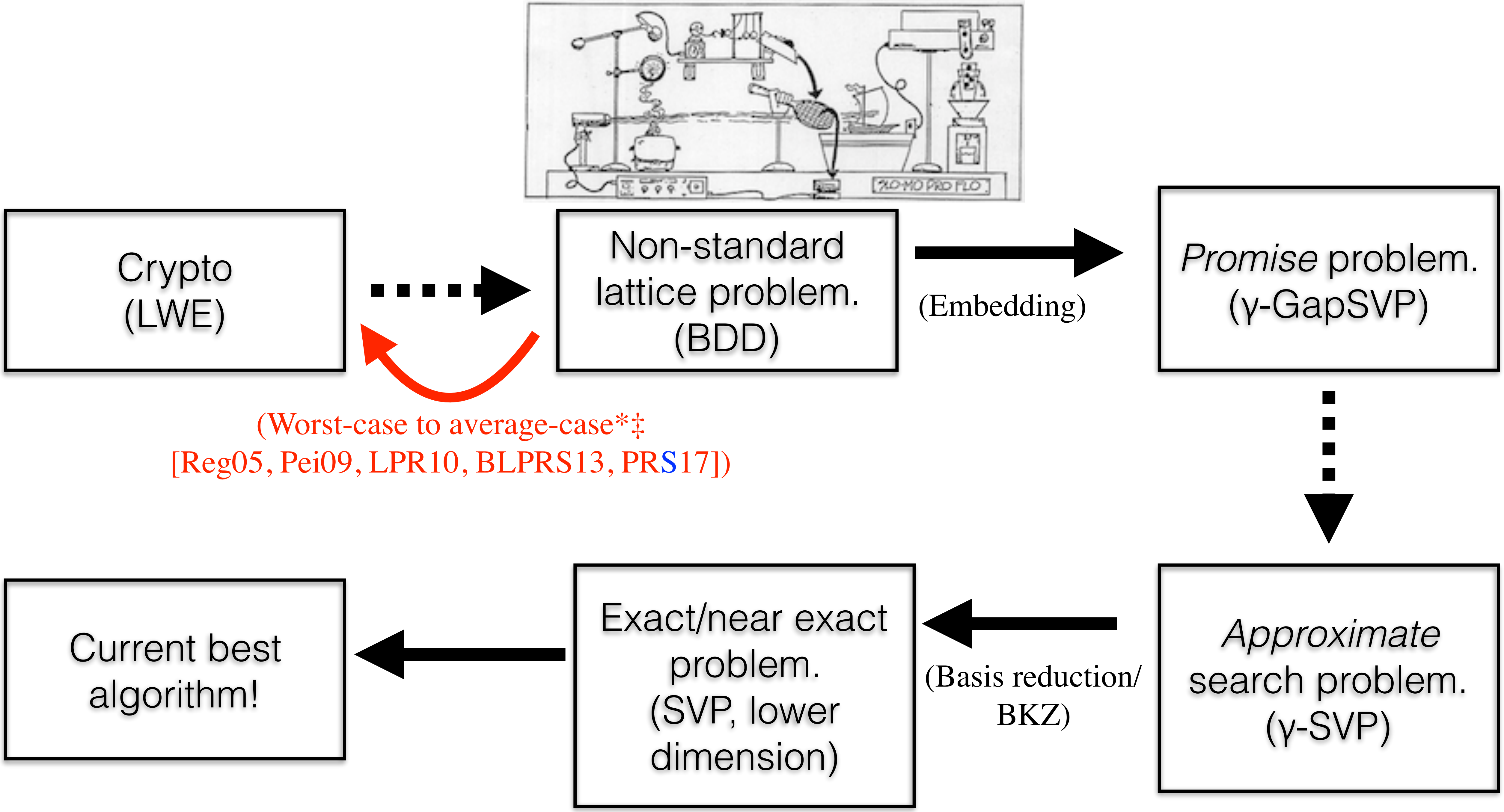
But maybe we can prove *something*?

# Security of Lattice-Based Crypto





# Security of Lattice-Based Crypto



Crypto (LWE)

Non-standard lattice problem. (BDD)

(Worst-case to average-case*‡ [Reg05, Pei09, LPR10, BLPRS13, PRS17])

(Embedding)

*Promise* problem. (γ-GapSVP)

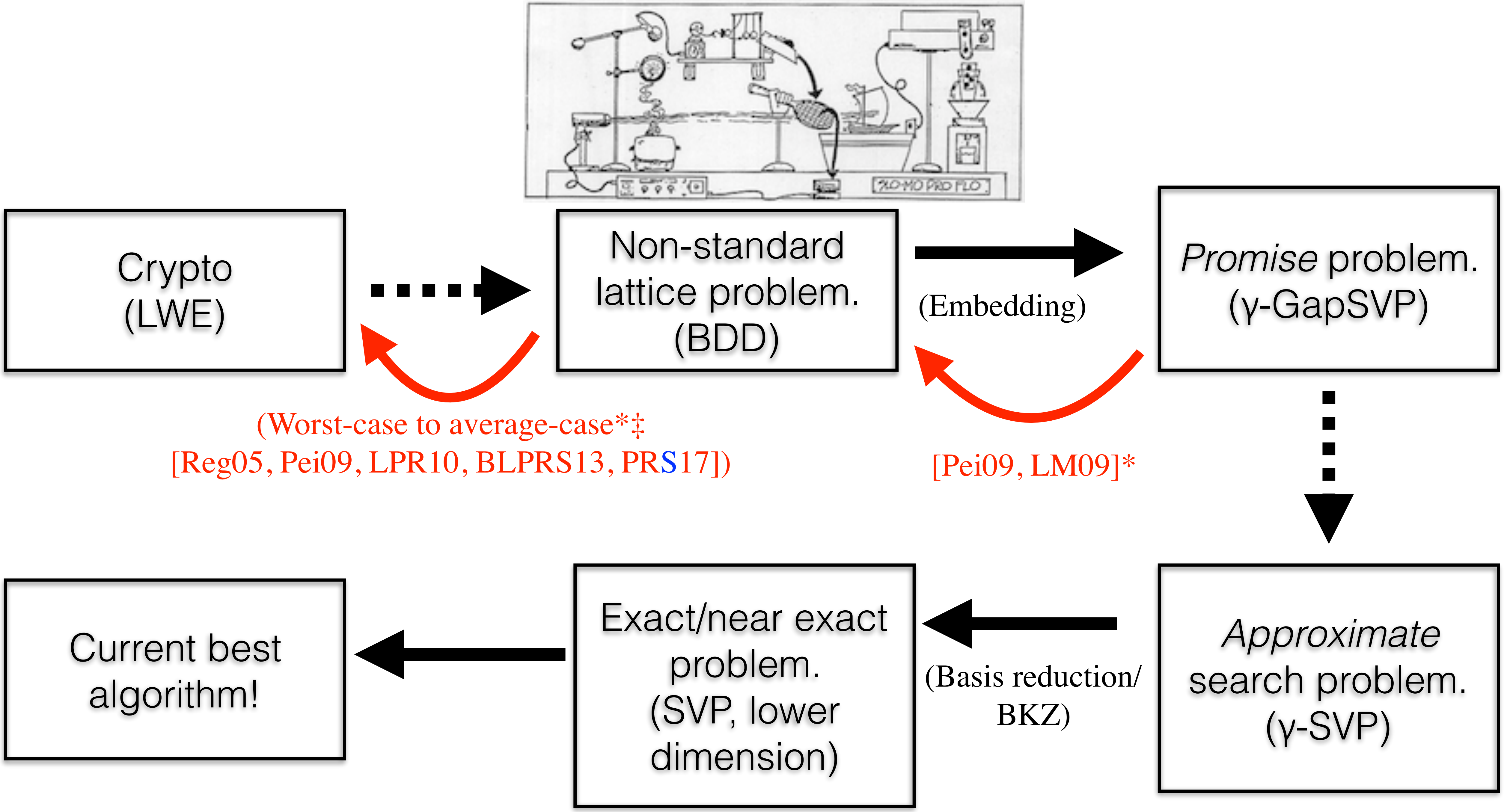
*Approximate* search problem. (γ-SVP)

(Basis reduction/ BKZ)

Exact/near exact problem. (SVP, lower dimension)

Current best algorithm!

# Security of Lattice-Based Crypto



Crypto (LWE)

Non-standard lattice problem. (BDD)

(Embedding)

*Promise* problem. (γ-GapSVP)

(Worst-case to average-case*‡ [Reg05, Pei09, LPR10, BLPRS13, PRS17])

[Pei09, LM09]*

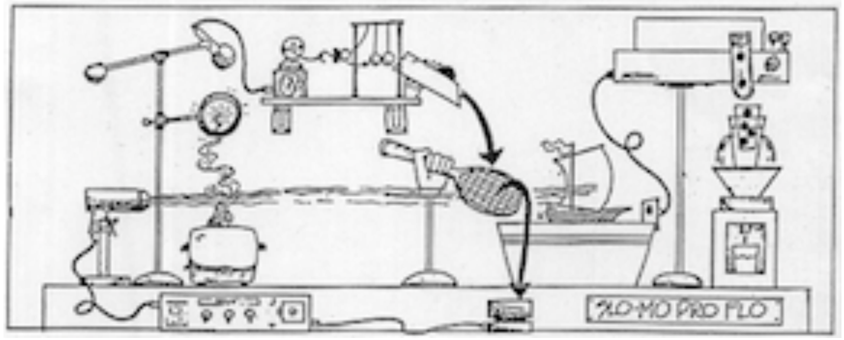
*Approximate* search problem. (γ-SVP)

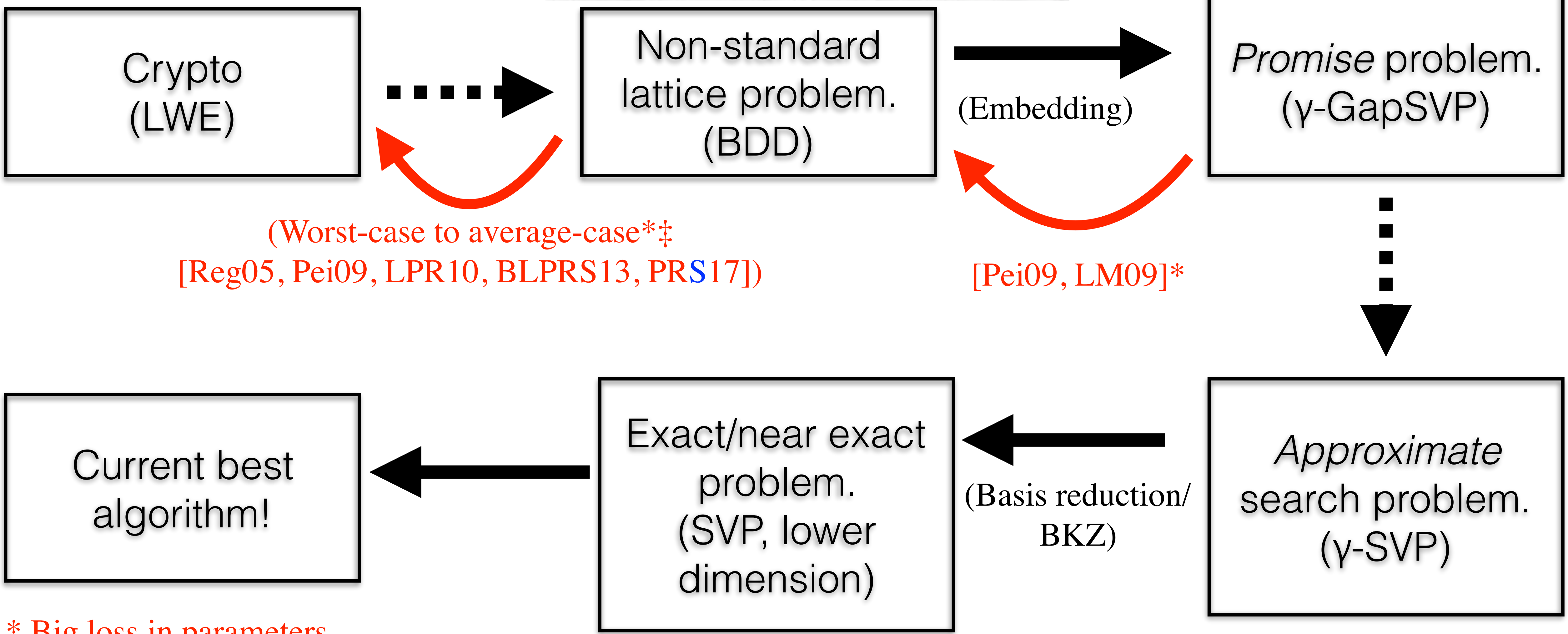
(Basis reduction/ BKZ)

Exact/near exact problem. (SVP, lower dimension)

Current best algorithm!

# Security of Lattice-Based Crypto





Crypto (LWE)

Non-standard lattice problem. (BDD)

(Embedding)

*Promise* problem. (γ-GapSVP)

(Worst-case to average-case*‡ [Reg05, Pei09, LPR10, BLPRS13, PRS17])

[Pei09, LM09]*

*Approximate* search problem. (γ-SVP)

(Basis reduction/ BKZ)

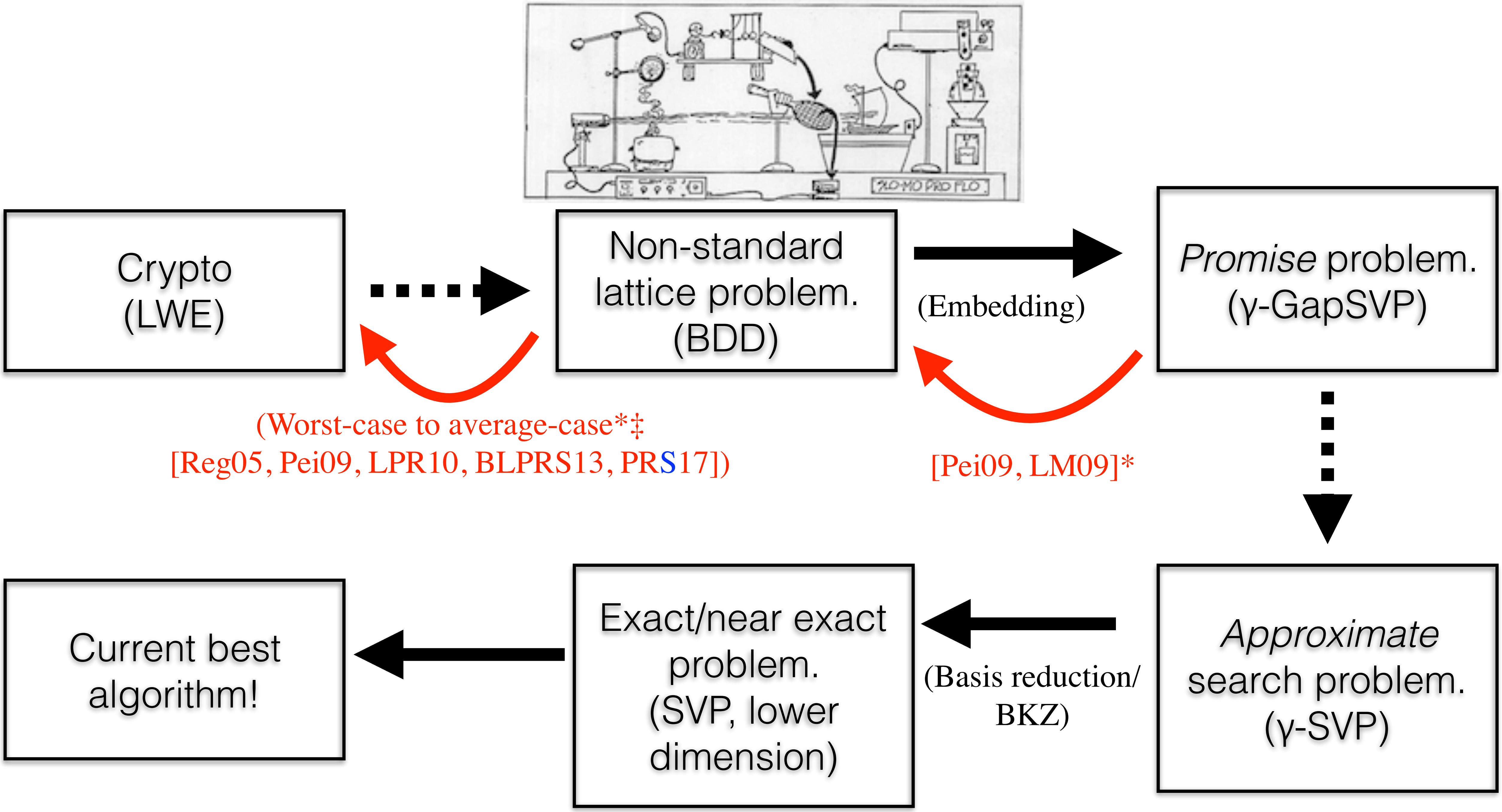
Exact/near exact problem. (SVP, lower dimension)

Current best algorithm!

* Big loss in parameters.

‡ Doesn't apply for many practical schemes.

# Security of Lattice-Based Crypto



Crypto (LWE)

Non-standard lattice problem. (BDD)

(Embedding)

*Promise* problem. (γ-GapSVP)

(Worst-case to average-case*‡ [Reg05, Pei09, LPR10, BLPRS13, PRS17])

[Pei09, LM09]*

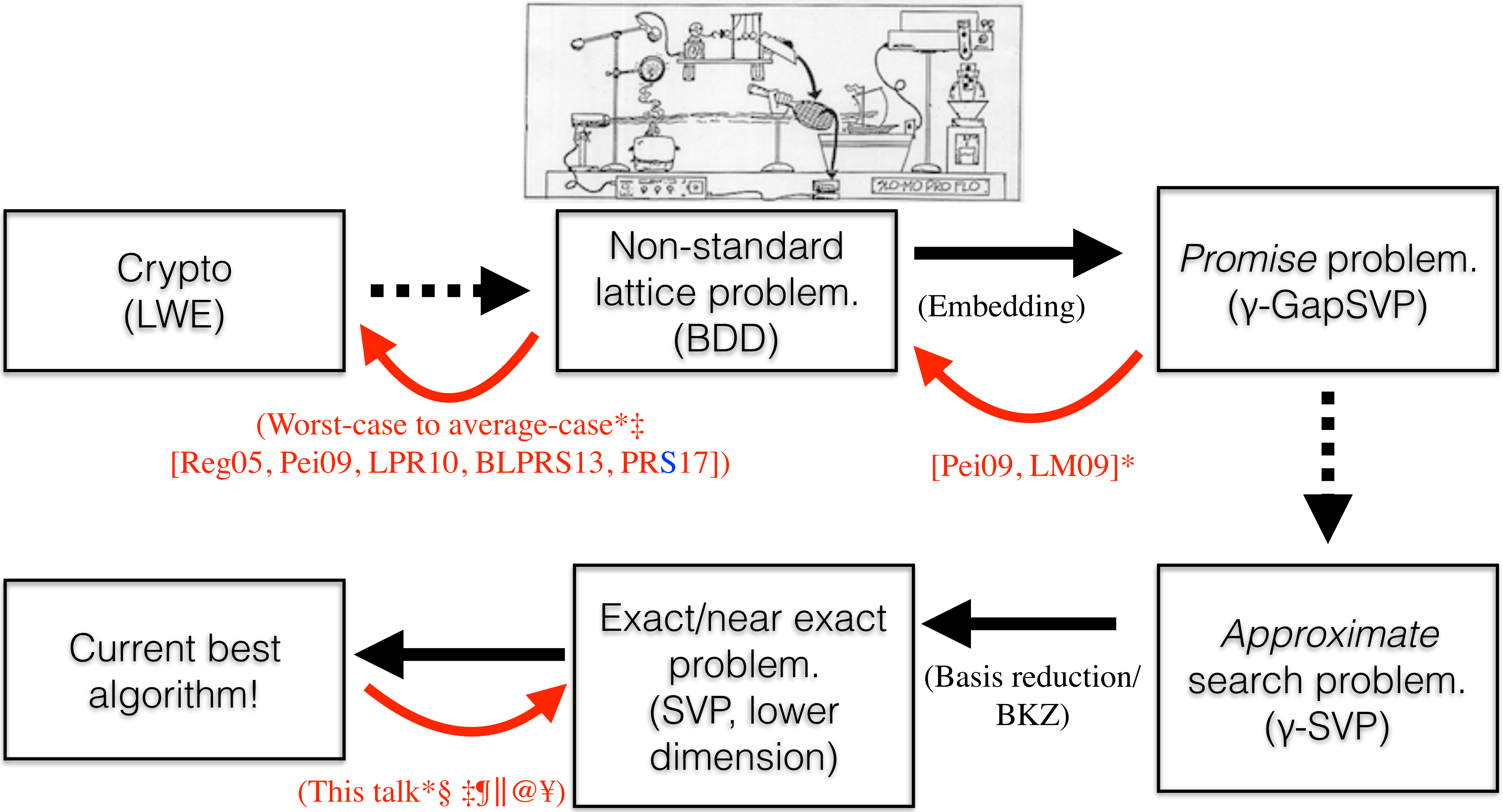
*Approximate* search problem. (γ-SVP)

(Basis reduction/ BKZ)

Exact/near exact problem. (SVP, lower dimension)

Current best algorithm!

# Security of Lattice-Based Crypto



Crypto (LWE)

Non-standard lattice problem. (BDD)

(Embedding)

*Promise* problem. (γ-GapSVP)

(Worst-case to average-case*‡ [Reg05, Pei09, LPR10, BLPRS13, PRS17])

[Pei09, LM09]*

*Approximate* search problem. (γ-SVP)

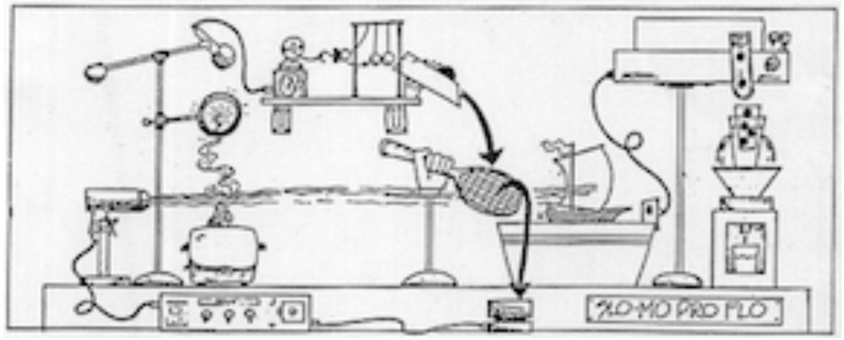
(Basis reduction/ BKZ)

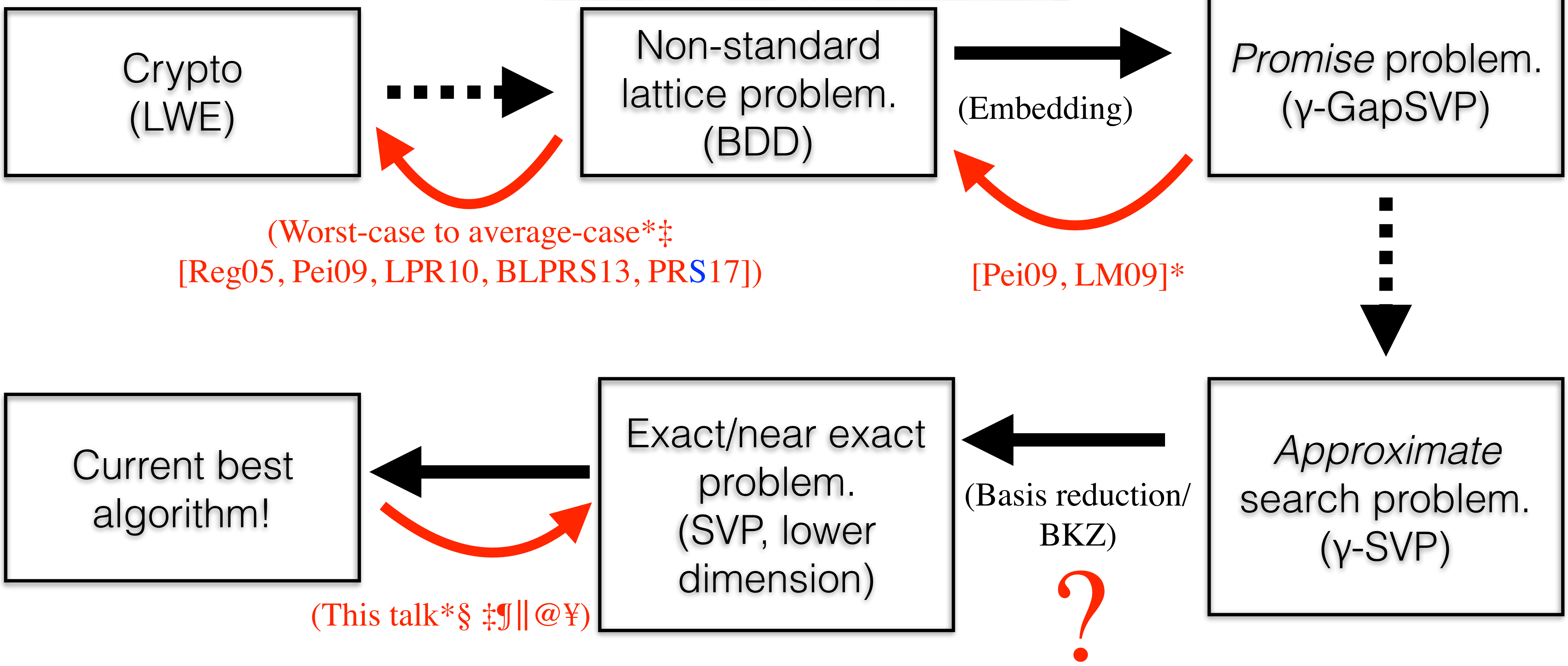
Exact/near exact problem. (SVP, lower dimension)

Current best algorithm!

(This talk*§ ‡¶‖@¥)

# Security of Lattice-Based Crypto





Crypto (LWE)

Non-standard lattice problem. (BDD)

(Embedding)

*Promise* problem. (γ-GapSVP)

(Worst-case to average-case*‡ [Reg05, Pei09, LPR10, BLPRS13, PRS17])

[Pei09, LM09]*

Current best algorithm!

Exact/near exact problem. (SVP, lower dimension)

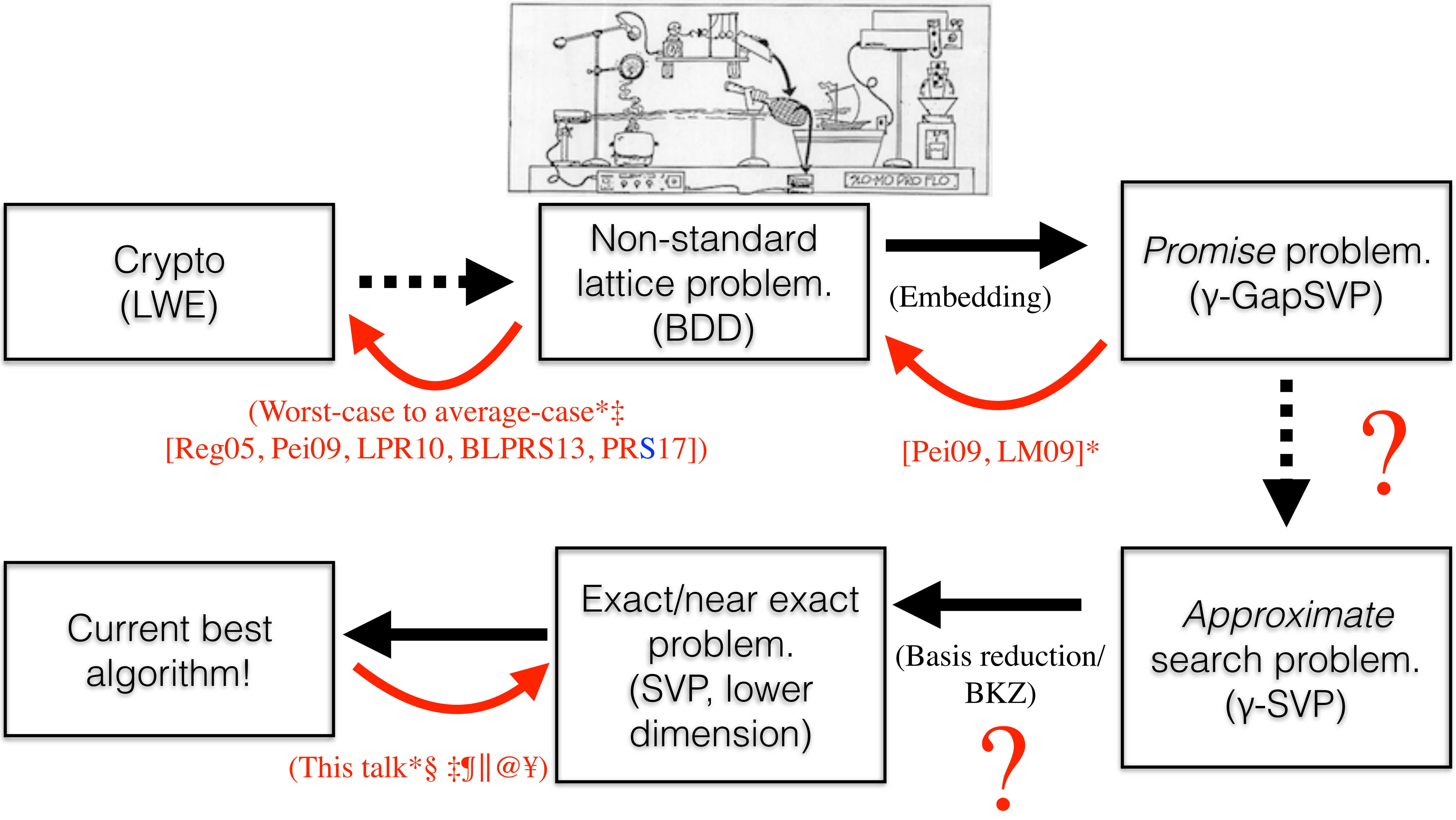
(Basis reduction/ BKZ)

?

*Approximate* search problem. (γ-SVP)

(This talk*§ ‡¶||@¥)

# Security of Lattice-Based Crypto



Crypto (LWE)

Non-standard lattice problem. (BDD)

(Worst-case to average-case*‡ [Reg05, Pei09, LPR10, BLPRS13, PRS17])

(Embedding)

*Promise* problem. (γ-GapSVP)

[Pei09, LM09]*

?

*Approximate* search problem. (γ-SVP)

(Basis reduction/ BKZ)

?

Exact/near exact problem. (SVP, lower dimension)

(This talk*§ ‡¶||@¥)

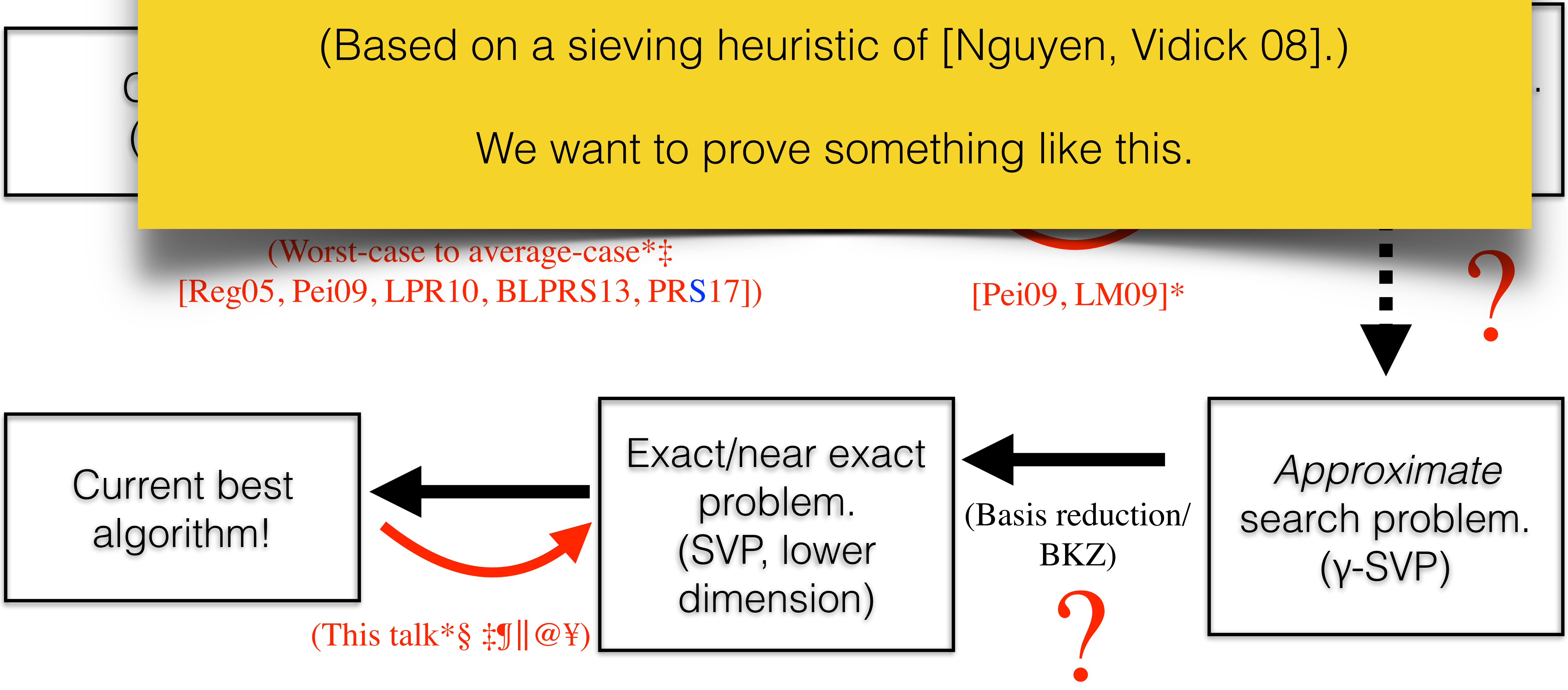
Current best algorithm!

# Security of Lattice-Based Crypto

Quantitative security estimates are (all?) based on the assumption that the fastest algorithm for exact/near exact SVP runs in time $(4/3)^{n/2}$.

(Based on a sieving heuristic of [Nguyen, Vidick 08].)

We want to prove something like this.



(Worst-case to average-case*‡ [Reg05, Pei09, LPR10, BLPRS13, PRS17])

[Pei09, LM09]*

?
Current best algorithm!
Exact/near exact problem. (SVP, lower dimension)
(Basis reduction/ BKZ)
?
*Approximate* search problem. (γ-SVP)
(This talk*§ ‡¶||@¥)

# Results
(Spoilers)

# Results (Spoilers)

- Lower bound of $2^{n+o(n)}$ time for **CVP** in “almost all” $\ell_p$ norms, **not including** $p = 2$ .

# Results
# (Spoilers)

- Lower bound of $2^{n+o(n)}$ time for **CVP** in “almost all” $\ell_p$ norms, **not including** $p = 2$ .
  - $2^{\Omega(n)}$ for all p, even for approximate CVP.

# Results (Spoilers)

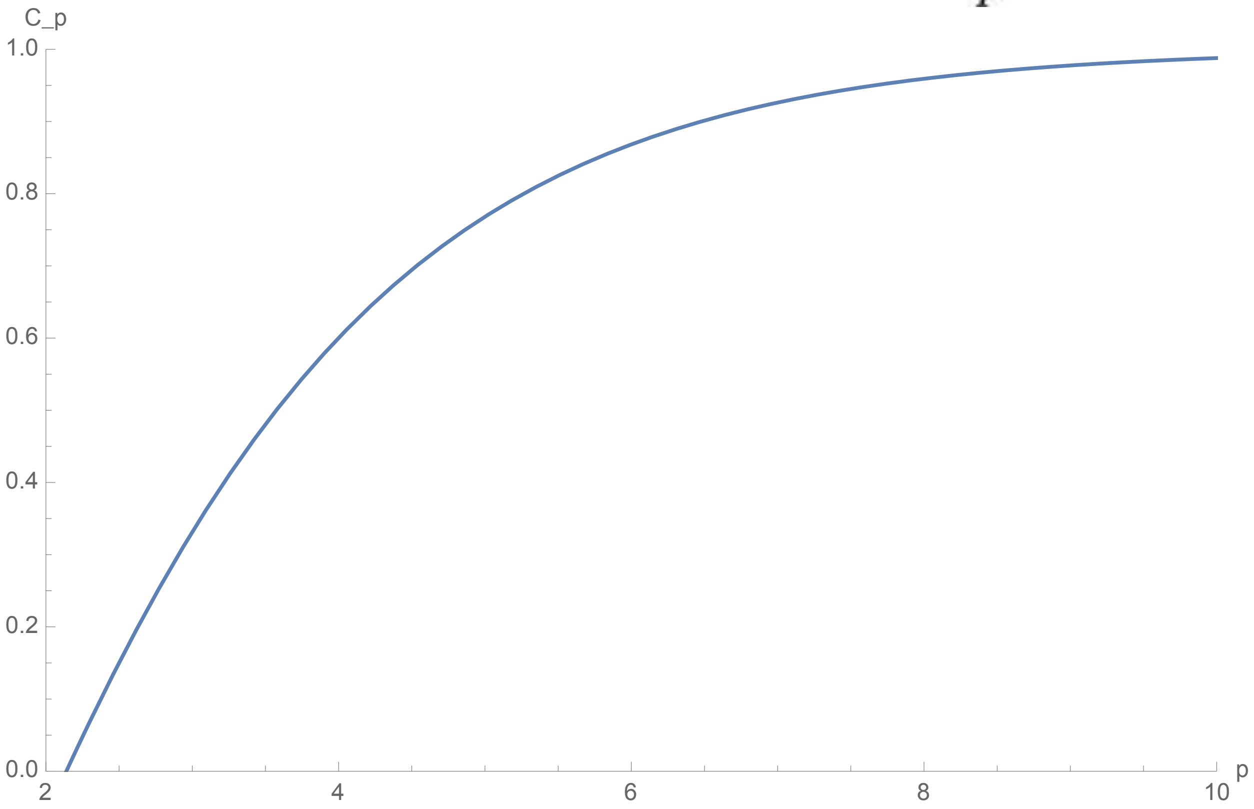
- Lower bound of $2^{n+o(n)}$ time for **CVP** in “almost all” $\ell_p$ norms, **not including** $p = 2$ .
  - $2^{\Omega(n)}$ for all p, even for approximate CVP.
  - Compare with the $2^{n-o(n)}$ -time algorithm for $p = 2$.

# Results (Spoilers)

- Lower bound of $2^{n+o(n)}$ time for **CVP** in "almost all" $\ell_p$ norms, **not including** $p = 2$ .
  - $2^{\Omega(n)}$ for all p, even for approximate CVP.
  - Compare with the $2^{n-o(n)}$ -time algorithm for $p = 2$ .
- Lower bound of $2^{C_p n}$ time for SVP in "almost all" $\ell_p$ norms with $p \gtrsim 2.14$ .

# Results (Spoilers)

- Lower bound of $2^{n+o(n)}$ time for **CVP** in "almost all" $\ell_p$ norms, **not including** $p = 2$ .
  - $2^{\Omega(n)}$ for all p, even for approximate CVP.
  - Compare with the $2^{n-o(n)}$ -time algorithm for $p = 2$ .
- Lower bound of $2^{C_p n}$ time for SVP in "almost all" $\ell_p$ norms with $p \gtrsim 2.14$ .



# Results (Spoilers)

- Lower bound of $2^{n+o(n)}$ time for **CVP** in "almost all" $\ell_p$ norms, **not including** $p = 2$ .
  - $2^{\Omega(n)}$ for all p, even for approximate CVP.
  - Compare with the $2^{n-o(n)}$ -time algorithm for $p = 2$ .
- Lower bound of $2^{C_p n}$ time for SVP in "almost all" $\ell_p$ norms with $p \gtrsim 2.14$ .

# Results (Spoilers)

- Lower bound of $2^{n+o(n)}$ time for **CVP** in "almost all" $\ell_p$ norms, **not including** $p = 2$ .
  - $2^{\Omega(n)}$ for all p, even for approximate CVP.
  - Compare with the $2^{n-o(n)}$ -time algorithm for $p = 2$.

- Lower bound of $2^{C_p n}$ time for SVP in "almost all" $\ell_p$ norms with $p \gtrsim 2.14$.

- Lower bound of $2^{\Omega(n)}$ for SVP for all p.

# Results (Spoilers)

- Lower bound of $2^{n+o(n)}$ time for **CVP** in “almost all” $\ell_p$ norms, **not including** $p = 2$ .
    - $2^{\Omega(n)}$ for all p, even for approximate CVP.
    - Compare with the $2^{n-o(n)}$ -time algorithm for $p = 2$ .

- Lower bound of $2^{C_p n}$ time for SVP in “almost all” $\ell_p$ norms with $p \gtrsim 2.14$ .

- Lower bound of $2^{\Omega(n)}$ for SVP for all p.
    - (This result is meant to surprise you later…)

# Results (Spoilers)

- Lower bound of $2^{n+o(n)}$ time for **CVP** in “almost all” $\ell_p$ norms, **not including** $p = 2$ .
  - $2^{\Omega(n)}$ for all p, even for approximate CVP.
  - Compare with the $2^{n-o(n)}$ -time algorithm for $p = 2$.

- Lower bound of $2^{C_p n}$ time for SVP in “almost all” $\ell_p$ norms with $p \gtrsim 2.14$.

- Lower bound of $2^{\Omega(n)}$ for SVP for all p.
  - (This result is meant to surprise you later…)
  - Compare with the $(4/3)^{n/2}$ heuristic lower bound.

# Act 2:
# Fine-Grained Hardness of CVP

Huck
Bennett

Alexander
Golovnev

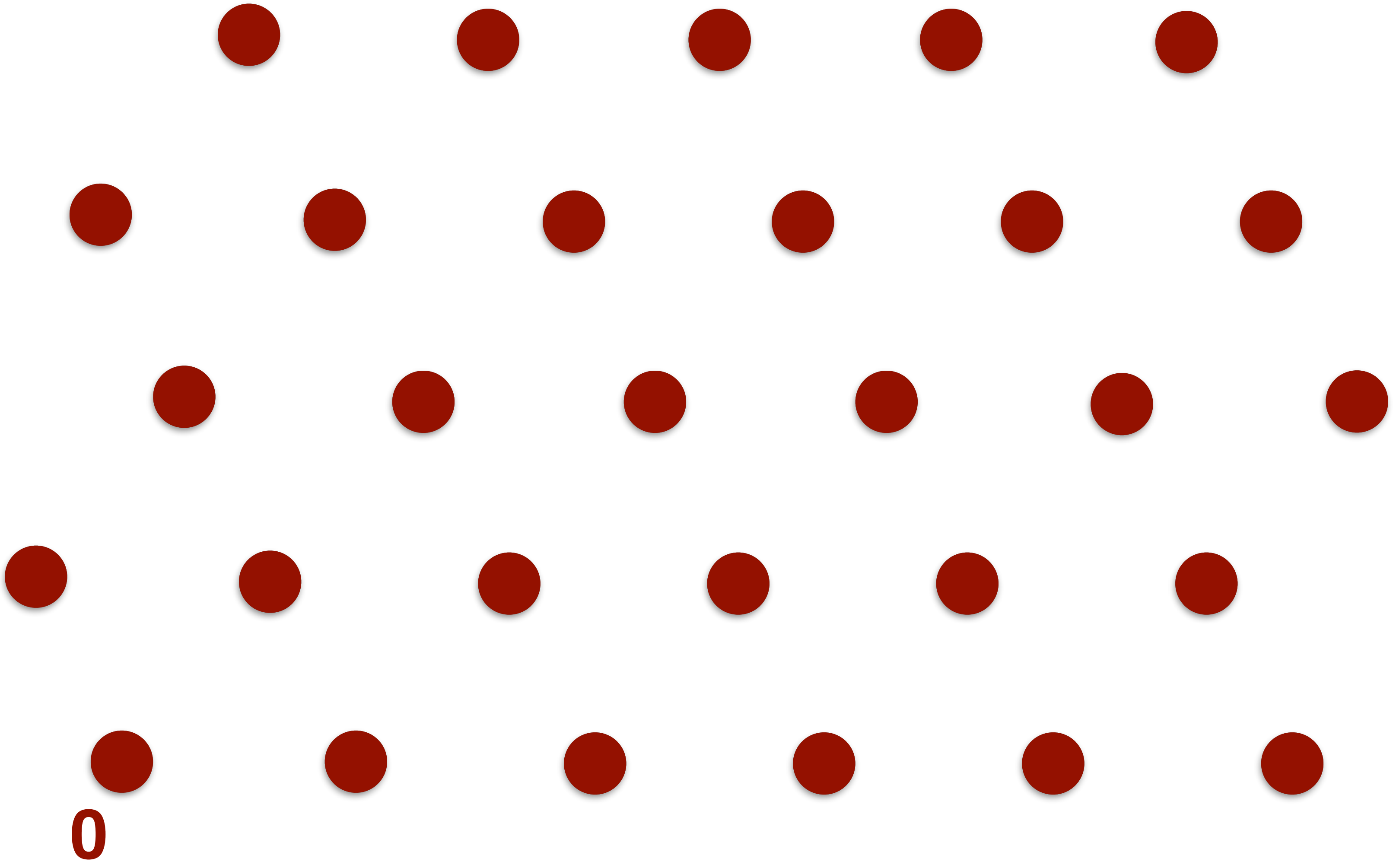
# Lattices

# Lattices

- $\mathcal{L}$ is a discrete set of vectors in $\mathbb{R}^d$.

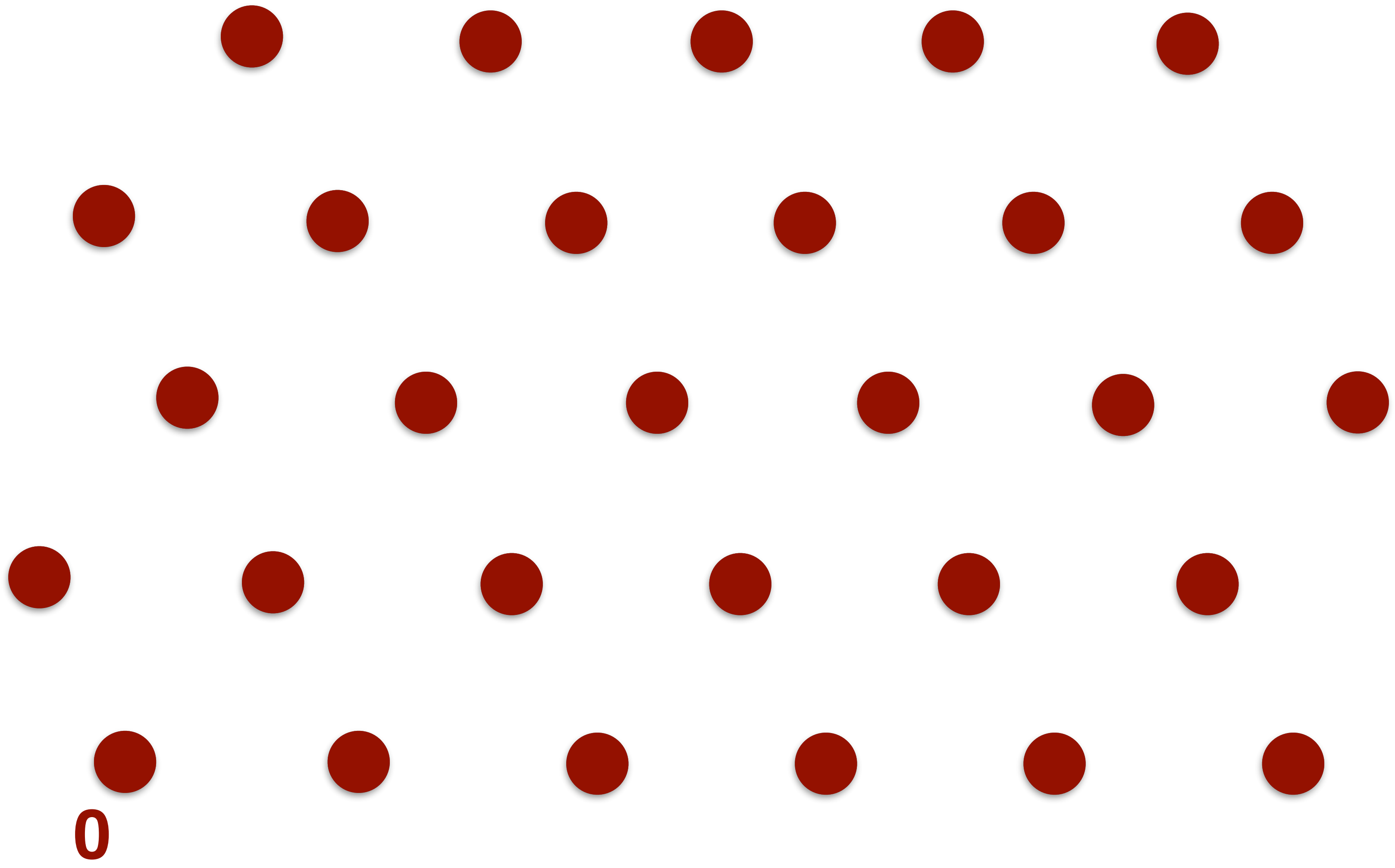
# Lattices

- $\mathcal{L}$ is a discrete set of vectors in $\mathbb{R}^d$.



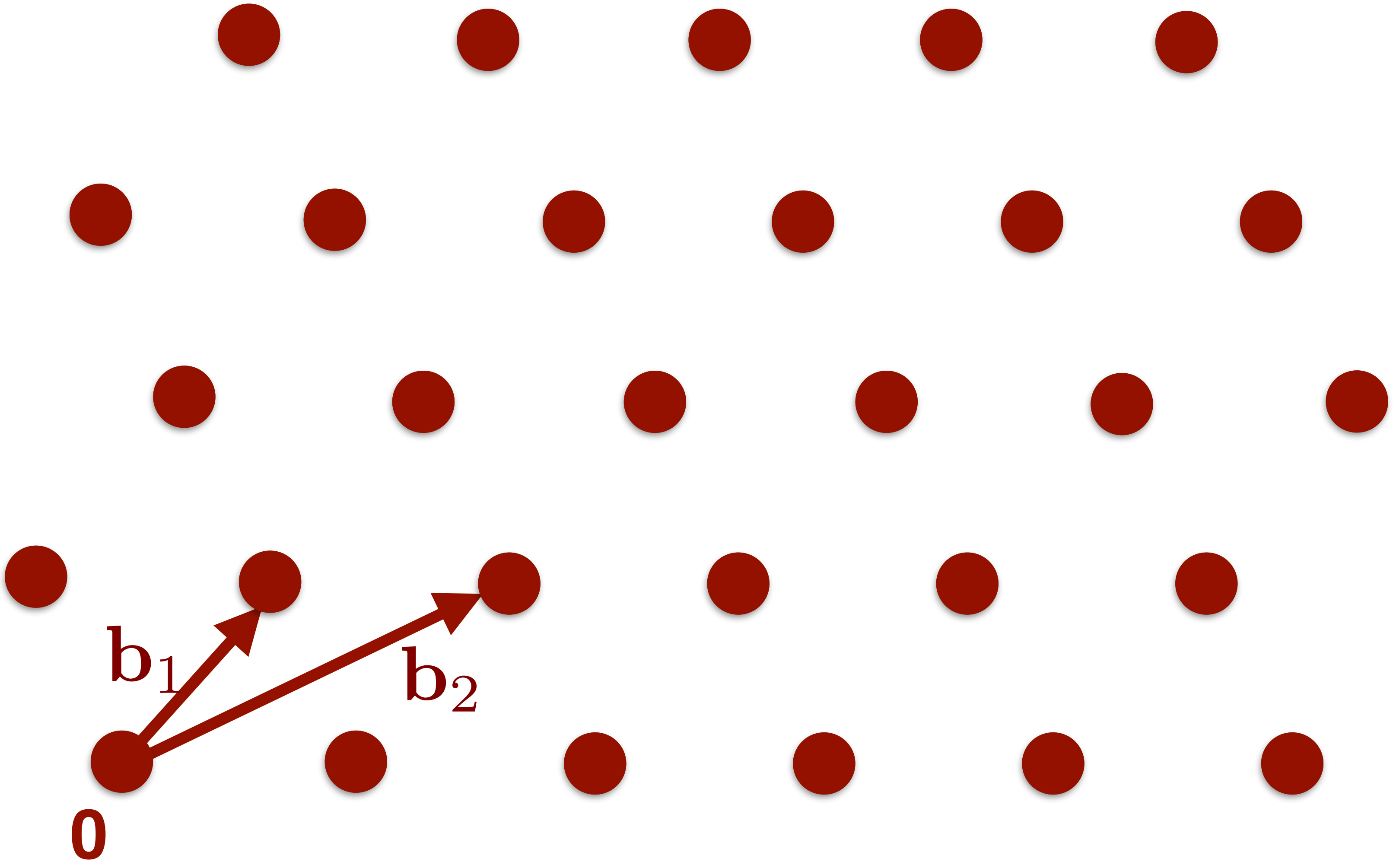
# Lattices

- $\mathcal{L}$ is a discrete set of vectors in $\mathbb{R}^d$.
- Specified by a basis $\mathbf{b}_1, \ldots, \mathbf{b}_n$, linearly independent vectors



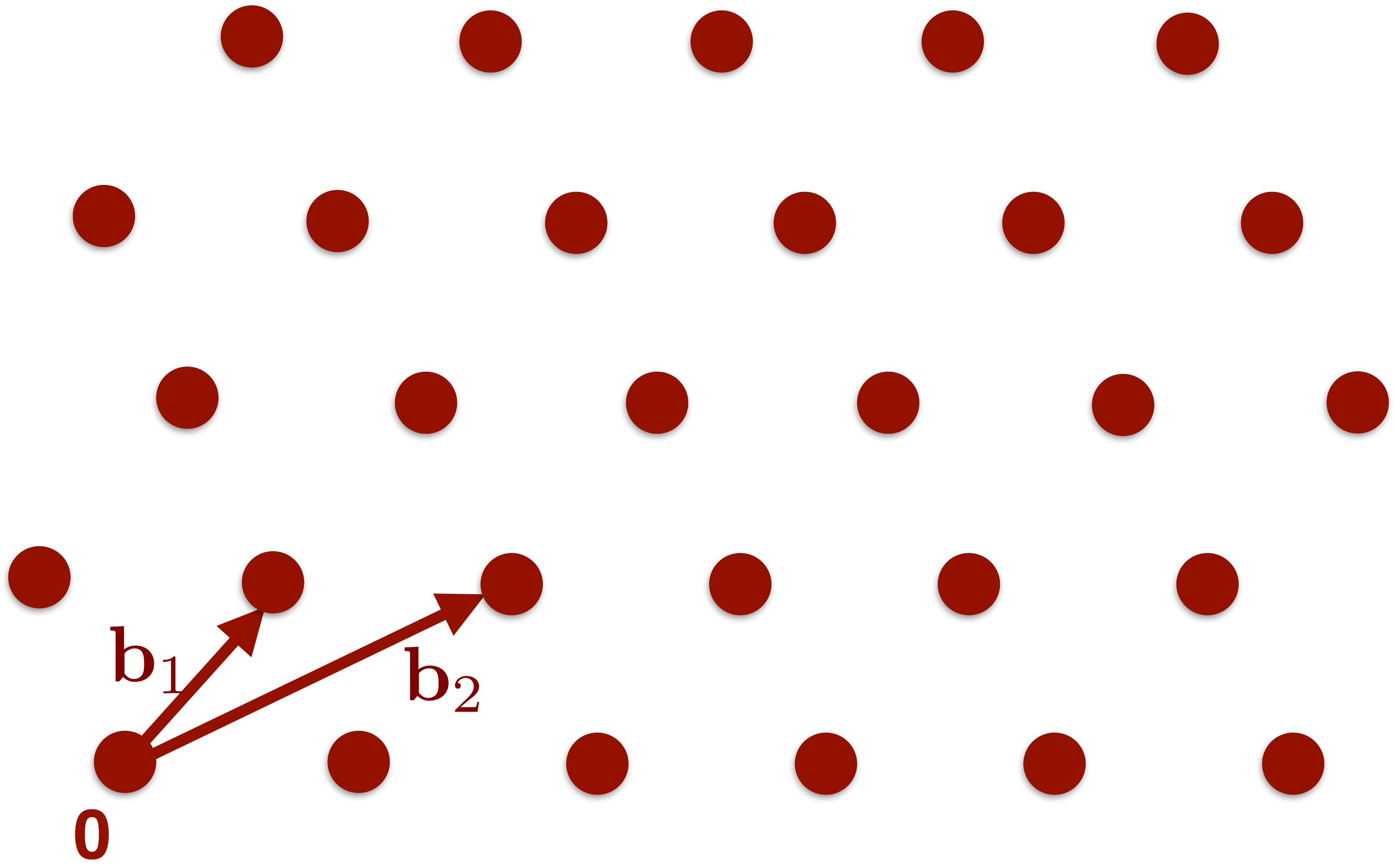
# Lattices

- $\mathcal{L}$ is a discrete set of vectors in $\mathbb{R}^d$.
- Specified by a basis $\mathbf{b}_1, \ldots, \mathbf{b}_n$, linearly independent vectors



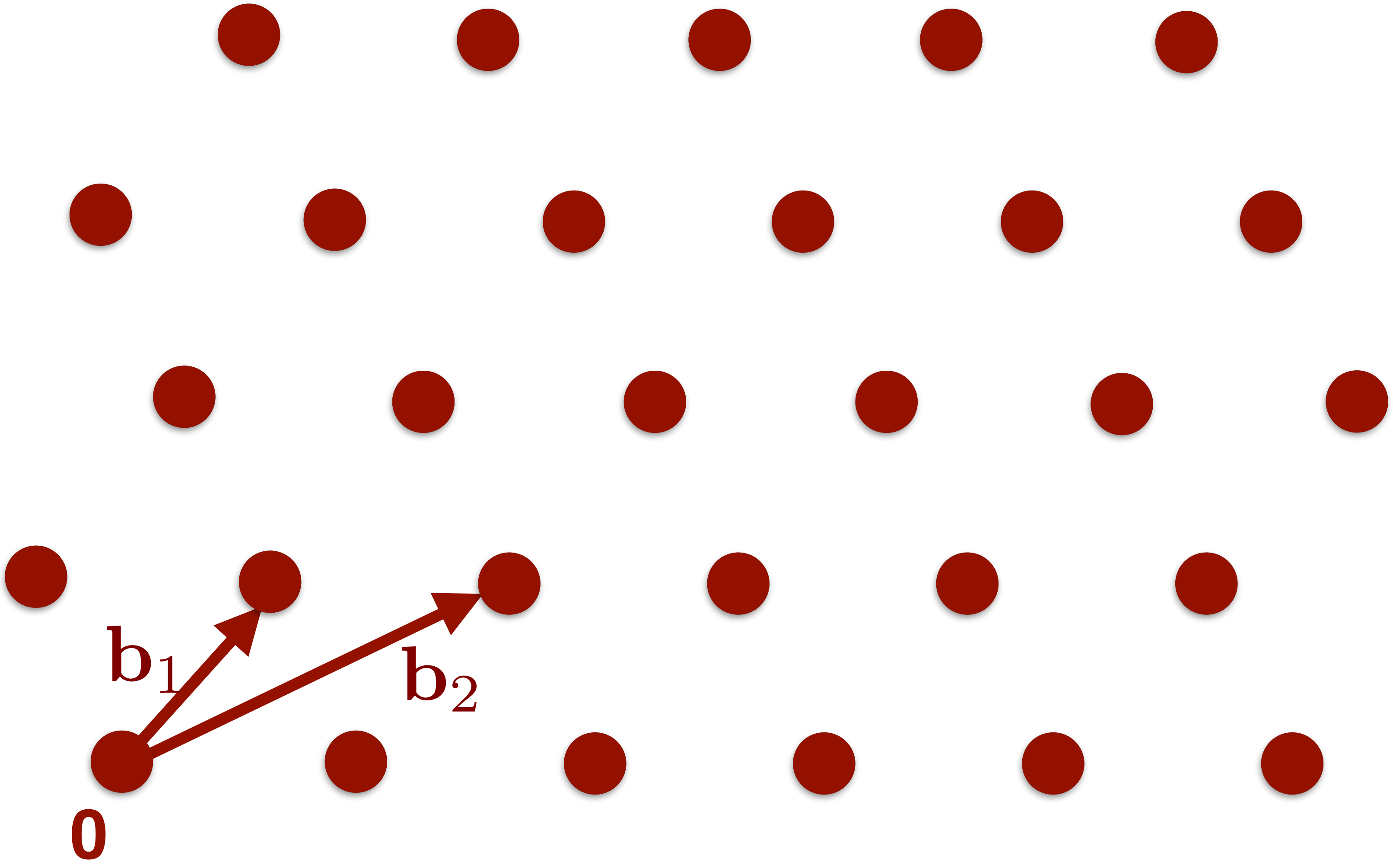
# Lattices

- $\mathcal{L}$ is a discrete set of vectors in $\mathbb{R}^d$.
- Specified by a basis $\mathbf{b}_1, \ldots, \mathbf{b}_n$, linearly independent vectors
- $\mathcal{L} = \{a_1\mathbf{b}_1 + \cdots + a_n\mathbf{b}_n \mid a_i \in \mathbb{Z}\}$.

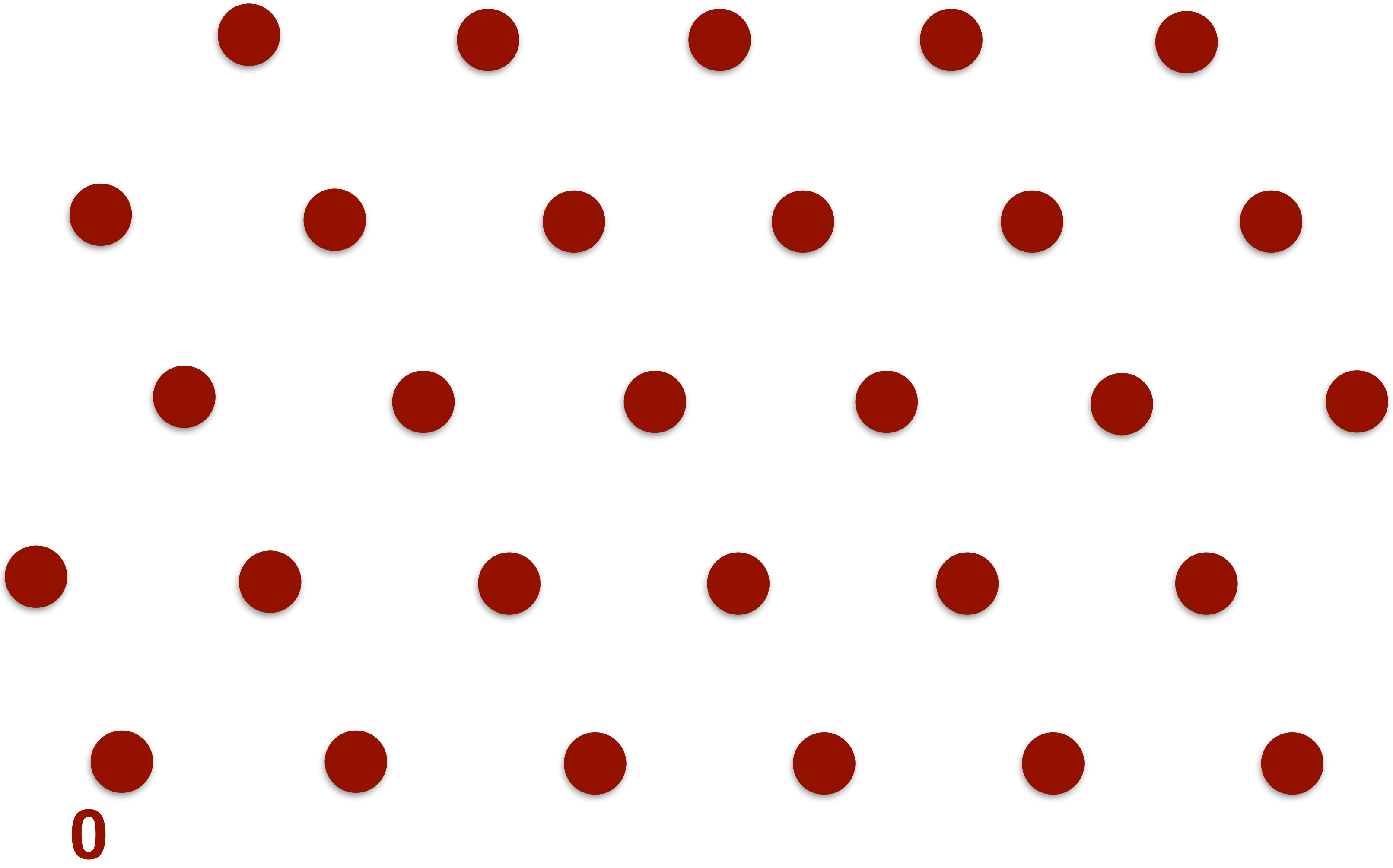


# Lattices

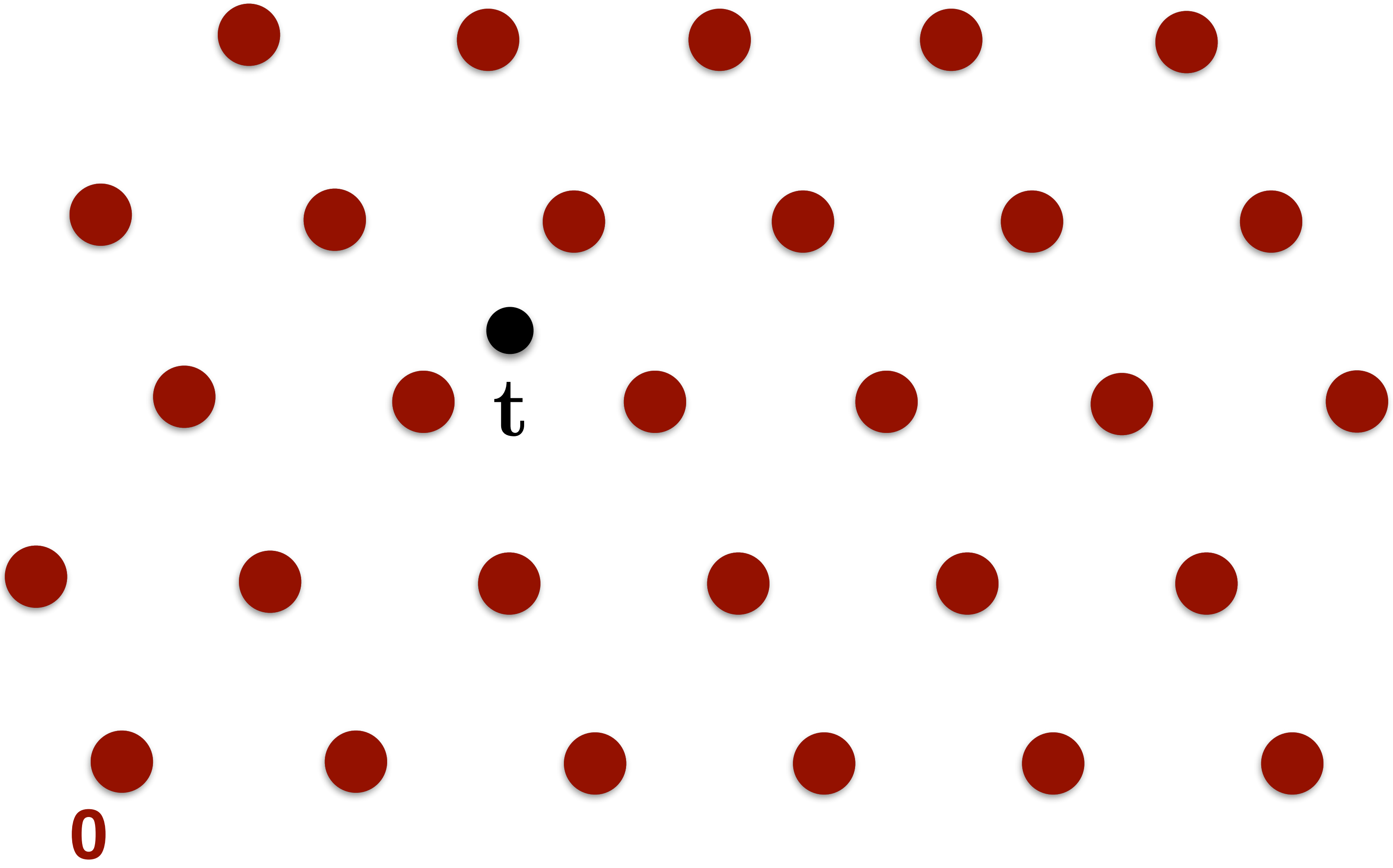
- $\mathcal{L}$ is a discrete set of vectors in $\mathbb{R}^d$.
- Specified by a basis $\mathbf{b}_1, \ldots, \mathbf{b}_n$, linearly independent vectors
- $\mathcal{L} = \{a_1\mathbf{b}_1 + \cdots + a_n\mathbf{b}_n \mid a_i \in \mathbb{Z}\}$.
- $n$ is the *rank* of the lattice, and $d$ is the *ambient dimension*.



# The Closest Vector Problem

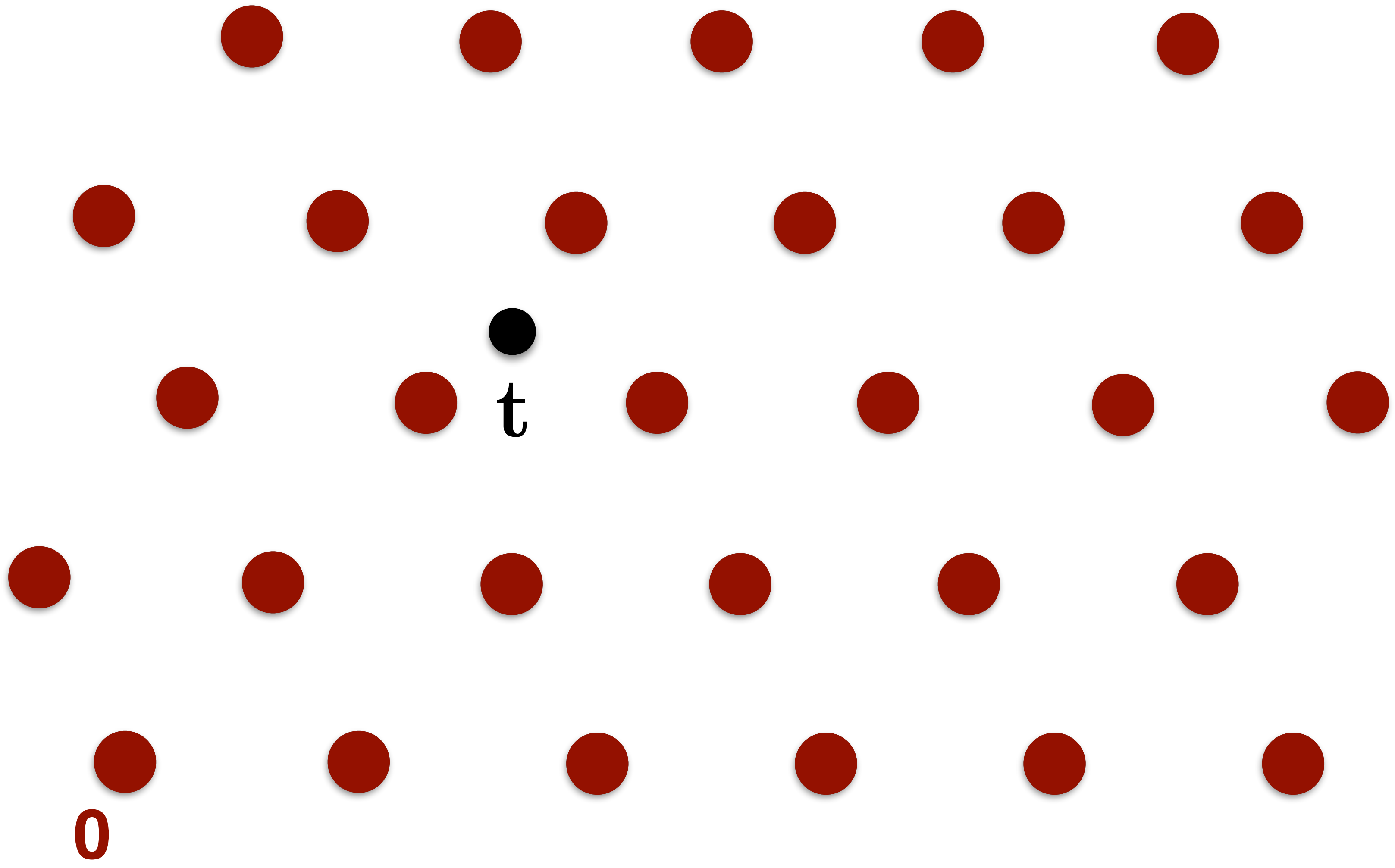


# The Closest Vector Problem



# The Closest Vector Problem

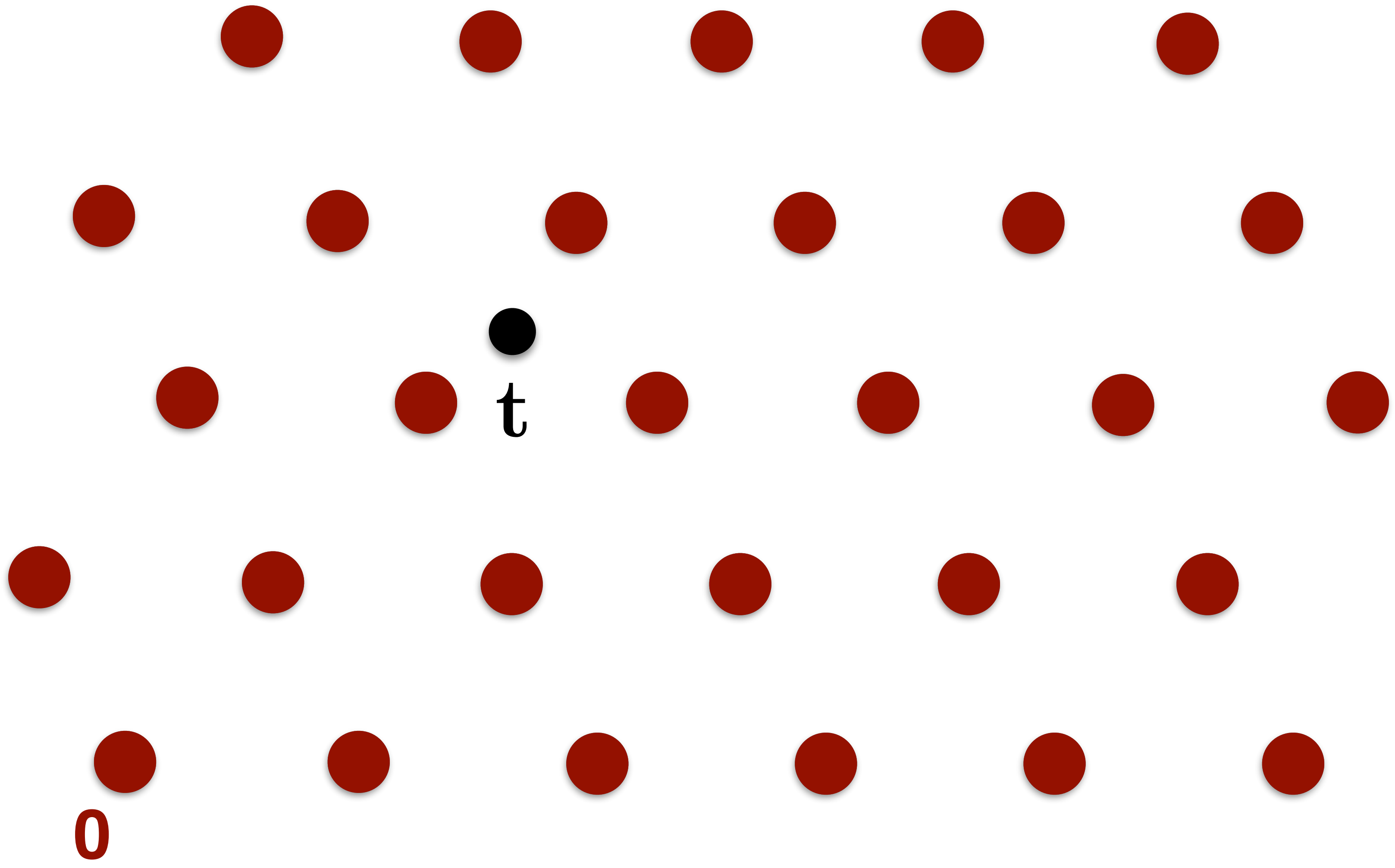
$$\mathrm{dist}_p(\mathbf{t}, \mathcal{L}) := \min_{\mathbf{y} \in \mathcal{L}} \|\mathbf{y} - \mathbf{t}\|_p$$



# The Closest Vector Problem

$$\mathrm{dist}_p(\mathbf{t}, \mathcal{L}) := \min_{\mathbf{y} \in \mathcal{L}} \|\mathbf{y} - \mathbf{t}\|_p$$

$$\|\mathbf{x}\|_p := (|x_1|^p + \cdots + |x_d|^p)^{1/p}$$

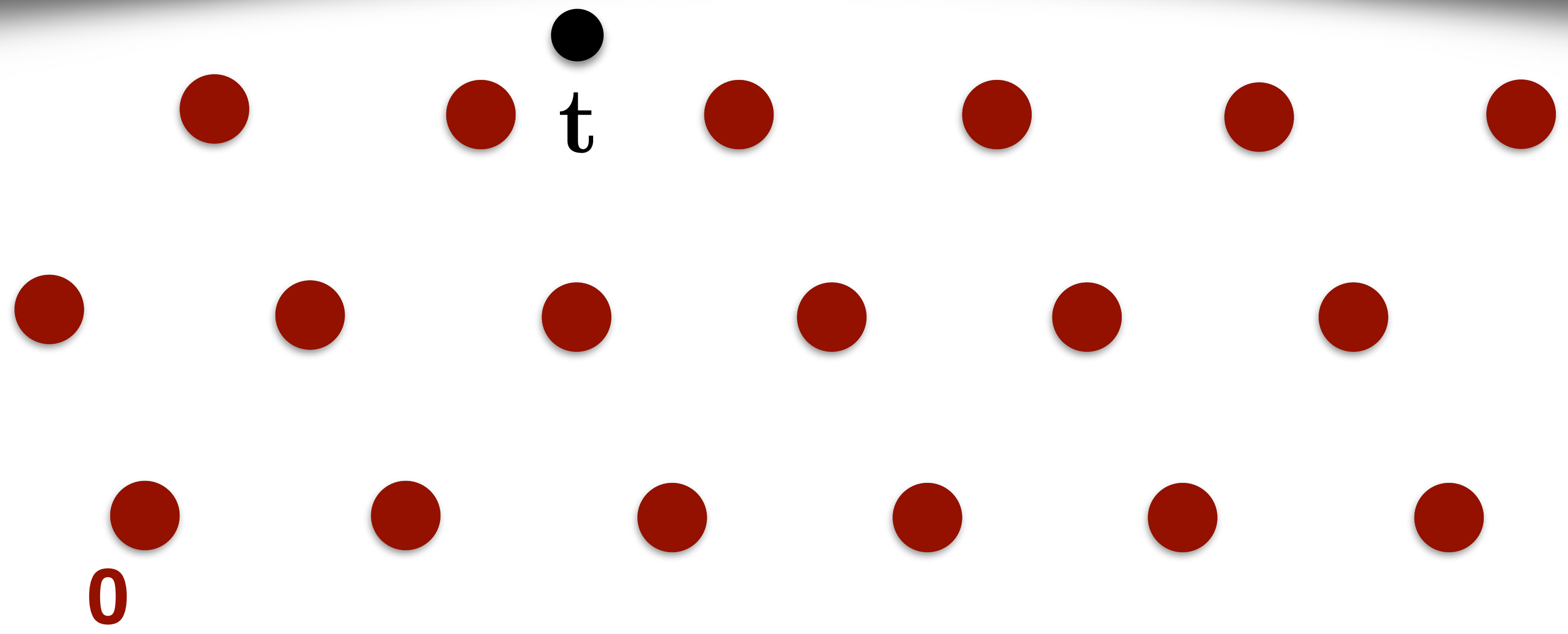


# The Closest Vector Problem

$$\mathrm{dist}_p(\mathbf{t}, \mathcal{L}) := \min_{\mathbf{y} \in \mathcal{L}} \|\mathbf{y} - \mathbf{t}\|_p$$

$$\|\mathbf{x}\|_p := (|x_1|^p + \cdots + |x_d|^p)^{1/p}$$

CVP$_p$ is the computational problem that asks us to compute $\mathrm{dist}_p(\mathbf{t}, \mathcal{L})$.

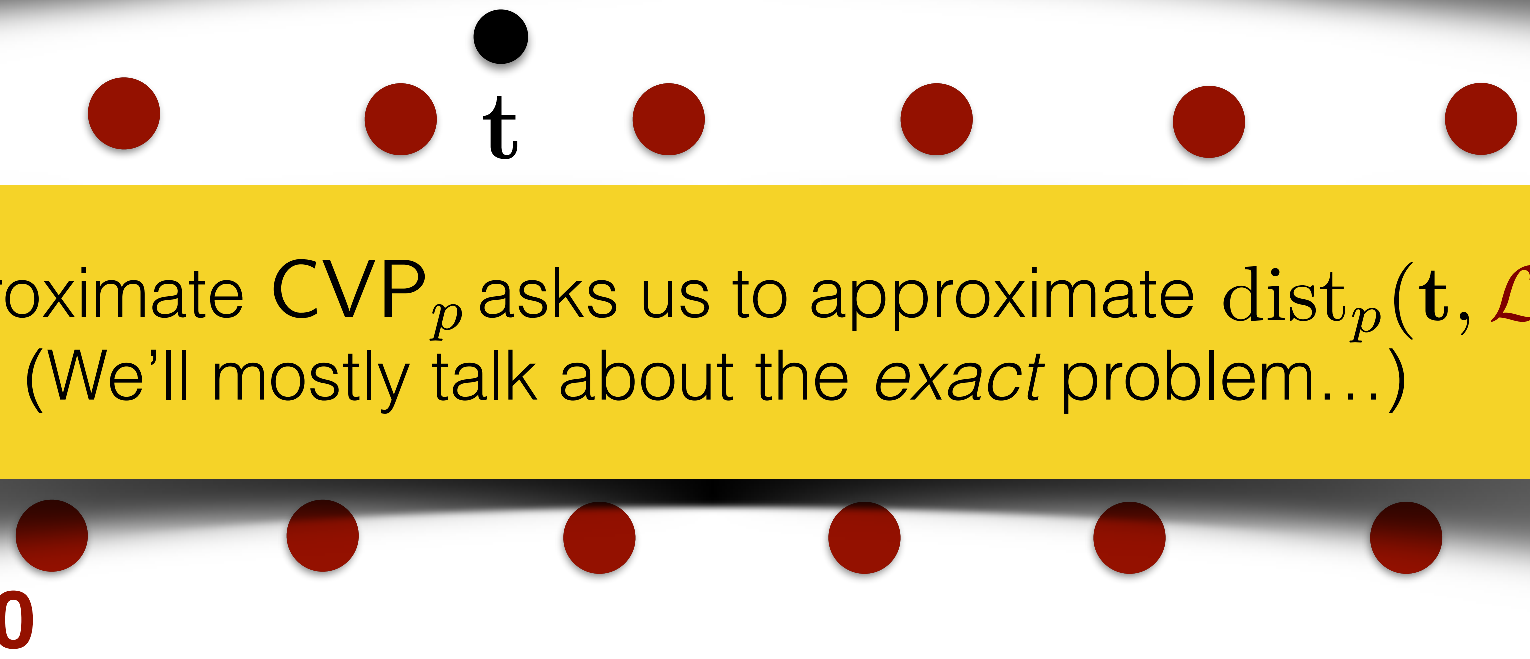


# The Closest Vector Problem

$$\mathrm{dist}_p(\mathbf{t}, \mathcal{L}) := \min_{\mathbf{y} \in \mathcal{L}} \|\mathbf{y} - \mathbf{t}\|_p$$

$$\|\mathbf{x}\|_p := (|x_1|^p + \cdots + |x_d|^p)^{1/p}$$

CVP$_p$ is the computational problem that asks us to compute $\mathrm{dist}_p(\mathbf{t}, \mathcal{L})$.



Approximate CVP$_p$ asks us to approximate $\mathrm{dist}_p(\mathbf{t}, \mathcal{L})$.
(We'll mostly talk about the *exact* problem…)

# The Closest Vector Problem

# The Closest Vector Problem

# The Closest Vector Problem

- $\mathrm{CVP}_2$ can be solved in time $2^{n+o(n)}$ [MV10, ADS15].

# The Closest Vector Problem

- $\mathrm{CVP}_2$ can be solved in time $2^{n+o(n)}$ [MV10, ADS15].
- $\mathrm{CVP}_p$ can be solved in time $n^{O(n)}$ and approximated in time $2^{O(n)}$ for all $1 \leq p \leq \infty$ [AKS02, BN09, Dadush 12].

# The Closest Vector Problem

- $\mathrm{CVP}_2$ can be solved in time $2^{n+o(n)}$ [MV10, ADS15].
- $\mathrm{CVP}_p$ can be solved in time $n^{O(n)}$ and approximated in time $2^{O(n)}$ for all $1 \leq p \leq \infty$ [AKS02, BN09, Dadush 12].
- $\mathrm{CVP}_p$ is NP-hard for all $1 \leq p \leq \infty$ [van Emde Boaz '81].

# The Closest Vector Problem

- $\mathrm{CVP}_2$ can be solved in time $2^{n+o(n)}$ [MV10, ADS15].
- $\mathrm{CVP}_p$ can be solved in time $n^{O(n)}$ and approximated in time $2^{O(n)}$ for all $1 \leq p \leq \infty$ [AKS02, BN09, Dadush 12].
- $\mathrm{CVP}_p$ is NP-hard for all $1 \leq p \leq \infty$ [van Emde Boaz '81].
- Even hard to approximate up to $n^{c/\log\log n}$ [ABSS93, DKRS03].

# The Closest Vector Problem

- $\mathrm{CVP}_2$ can be solved in time $2^{n+o(n)}$ [MV10, ADS15].
- $\mathrm{CVP}_p$ can be solved in time $n^{O(n)}$ and approximated in time $2^{O(n)}$ for all $1 \leq p \leq \infty$ [AKS02, BN09, Dadush 12].
- $\mathrm{CVP}_p$ is NP-hard for all $1 \leq p \leq \infty$ [van Emde Boaz '81].
- Even hard to approximate up to $n^{c/\log\log n}$ [ABSS93, DKRS03].
- "The hardest lattice problem."

# The Closest Vector Problem

- $\mathrm{CVP}_2$ can be solved in time $2^{n+o(n)}$ [MV10, ADS15].
- $\mathrm{CVP}_p$ can be solved in time $n^{O(n)}$ and approximated in time $2^{O(n)}$ for all $1 \le p \le \infty$ [AKS02, BN09, Dadush 12].
- $\mathrm{CVP}_p$ is NP-hard for all $1 \le p \le \infty$ [van Emde Boaz ’81].
- Even hard to approximate up to $n^{c/\log\log n}$ [ABSS93, DKRS03].
- “The hardest lattice problem.”
  - (In practice seems much harder than SVP)

# The Closest Vector Problem

- $\mathrm{CVP}_2$ can be solved in time $2^{n+o(n)}$ [MV10, ADS15].
- $\mathrm{CVP}_p$ can be solved in time $n^{O(n)}$ and approximated in time $2^{O(n)}$ for all $1 \leq p \leq \infty$ [AKS02, BN09, Dadush 12].
- $\mathrm{CVP}_p$ is NP-hard for all $1 \leq p \leq \infty$ [van Emde Boaz '81].
- Even hard to approximate up to $n^{c/\log\log n}$ [ABSS93, DKRS03].
- "The hardest lattice problem."
  - (In practice seems much harder than SVP)
  - Hardness proofs for lattice problems (like SVP) typically go through CVP.

# The Closest Vector Problem

- $\mathrm{CVP}_2$ can be solved in time $2^{n+o(n)}$ [MV10, ADS15].
- $\mathrm{CVP}_p$ can be solved in time $n^{O(n)}$ and approximated in time $2^{O(n)}$ for all $1 \leq p \leq \infty$ [AKS02, BN09, Dadush 12].
- $\mathrm{CVP}_p$ is NP-hard for all $1 \leq p \leq \infty$ [van Emde Boaz ’81].
- Even hard to approximate up to $n^{c/\log\log n}$ [ABSS93, DKRS03].
- “The hardest lattice problem.”
  - (In practice seems much harder than SVP)
  - Hardness proofs for lattice problems (like SVP) typically go through CVP.

Real-world cryptographic applications require quantitative/fine-grained hardness.

Maybe there’s a $2^{\sqrt{n}}$ -time algorithm for CVP? Even a $2^{n/20}$-time would break crypto in practice.

We’ll show something close to $2^n$-time hardness of CVP.

# The Closest Vector Problem

- $\mathrm{CVP}_2$ can be solved in time $2^{n+o(n)}$ [MV10, ADS15].
- CVP $\leq p \leq \infty$

  [AK
- CVP
- Even
- “The
  - (In practice seems much harder than SVP)
  - Hardness proofs for lattice problems (like SVP) typically go through CVP.

Before this work, only fine-grained hardness result known was a $2^{\Omega(n)}$-time lower bound [folklore, Yeom15].

Real-world cryptographic applications require quantitative/fine-grained hardness.

Maybe there’s a $2^{\sqrt{n}}$ -time algorithm for CVP? Even a $2^{n/20}$-time would break crypto in practice.

We’ll show something close to $2^n$-time hardness of CVP.

# The Strong Exponential Time Hypothesis (SETH)

# The Strong Exponential Time Hypothesis (SETH)

k-SAT:

# The Strong Exponential Time Hypothesis (SETH)

k-SAT:

$$(x_1 \vee \overline{x}_7 \vee \cdots \vee \overline{x}_{72}) \wedge (\overline{x}_{103} \vee \overline{x}_2 \vee \cdots \vee x_{42}) \wedge \cdots \wedge (\overline{x}_5 \vee x_{17} \vee \cdots \vee x_{112})$$

# The Strong Exponential Time Hypothesis (SETH)

k-SAT:

$$(x_1 \vee \overline{x}_7 \vee \cdots \vee \overline{x}_{72}) \wedge \underbrace{(\overline{x}_{103} \vee \overline{x}_2 \vee \cdots \vee x_{42})}_{k \text{ literals per clause}} \wedge \cdots \wedge (\overline{x}_5 \vee x_{17} \vee \cdots \vee x_{112})$$

# The Strong Exponential Time Hypothesis (SETH)

k-SAT:

$$(x_1 \vee \overline{x}_7 \vee \cdots \vee \overline{x}_{72}) \wedge \underbrace{(\overline{x}_{103} \vee \overline{x}_2 \vee \cdots \vee x_{42})}_{k \text{ literals per clause}} \wedge \cdots \wedge (\overline{x}_5 \vee x_{17} \vee \cdots \vee x_{112})$$

$n$ variables, $m$ clauses

# The Strong Exponential Time Hypothesis (SETH)

k-SAT:

$$(x_1 \vee \overline{x}_7 \vee \cdots \vee \overline{x}_{72}) \wedge \underbrace{(\overline{x}_{103} \vee \overline{x}_2 \vee \cdots \vee x_{42})}_{k \text{ literals per clause}} \wedge \cdots \wedge (\overline{x}_5 \vee x_{17} \vee \cdots \vee x_{112})$$

$n$ variables, $m$ clauses

**Conjecture** (SETH, IP99)**.** *There exists a constant integer $k$ such that no algorithm can solve $k$-SAT in $2^{0.99n}$ time.*

# The Strong Exponential Time Hypothesis (SETH)

k-SAT:

$$(x_1 \vee \overline{x}_7 \vee \cdots \vee \overline{x}_{72}) \wedge \underbrace{(\overline{x}_{103} \vee \overline{x}_2 \vee \cdots \vee x_{42})}_{k \text{ literals per clause}} \wedge \cdots \wedge (\overline{x}_5 \vee x_{17} \vee \cdots \vee x_{112})$$

$n$ variables, $m$ clauses

**Conjecture** (SETH, IP99). *There exists a constant integer $k$ such that no algorithm can solve $k$-SAT in $2^{0.99n}$ time.*

We want to show a reduction from k-SAT on n variables to CVP on a lattice of rank n.

# 2-SAT
# (a very special case…)

# 2-SAT
# (a very special case…)

$$(x_1 \vee x_2) \wedge (\overline{x}_1 \vee x_3) \wedge (\overline{x}_2 \vee \overline{x}_3)$$

# 2-SAT
# (a very special case…)

$$(x_1 \vee x_2) \wedge (\overline{x}_1 \vee x_3) \wedge (\overline{x}_2 \vee \overline{x}_3)$$

$$\mathbf{B} := \left( \begin{array}{c} 2\Phi \\ \hline 2\alpha I_n \end{array} \right)$$

# 2-SAT
# (a very special case…)

$$(x_1 \vee x_2) \wedge (\overline{x}_1 \vee x_3) \wedge (\overline{x}_2 \vee \overline{x}_3)$$

$$\Phi \in \mathbb{R}^{m \times n}$$

$$\mathbf{B} := \left( \begin{array}{c} 2\Phi \\ \hline 2\alpha I_n \end{array} \right)$$

# 2-SAT (a very special case…)

$$(x_1 \vee x_2) \wedge (\overline{x}_1 \vee x_3) \wedge (\overline{x}_2 \vee \overline{x}_3)$$

$$\Phi \in \mathbb{R}^{m \times n}$$

$$\mathbf{B} := \left( \begin{array}{c} 2\Phi \\ \hline 2\alpha I_n \end{array} \right)$$

$$\mathbf{t} := \left( \begin{array}{c} t_1 \\ t_2 \\ \vdots \\ t_m \\ \hline \alpha \\ \alpha \\ \vdots \\ \alpha \end{array} \right)$$

# 2-SAT
# (a very special case…)

$$\Phi \in \mathbb{R}^{m \times n}$$

$$\mathbf{B} := \left( \begin{array}{c} 2\Phi \\ \hline \boxed{2\alpha I_n} \end{array} \right)$$

$$\mathbf{t} := \left( \begin{array}{c} t_1 \\ t_2 \\ \vdots \\ t_m \\ \hline \boxed{\begin{matrix} \alpha \\ \alpha \\ \vdots \\ \alpha \end{matrix}} \end{array} \right)$$

# 2-SAT
# (a very special case…)

$\Phi \in \mathbb{R}^{m \times n}$

$$\mathbf{B} := \left(\begin{array}{c} 2\Phi \\ \hline \boxed{2\alpha I_n} \end{array}\right)$$

$$\begin{array}{r} \\ \\ \\ \\ \mathbf{t} := \\ \\ \boldsymbol{\alpha} := \\ \\ \\ \end{array} \left(\begin{array}{c} t_1 \\ t_2 \\ \vdots \\ t_m \\ \hline \boxed{\begin{array}{c} \alpha \\ \alpha \\ \vdots \\ \alpha \end{array}} \end{array}\right)$$

# 2-SAT
# (a very special case…)

$$\Phi \in \mathbb{R}^{m \times n}$$

$$\mathbf{B} := \begin{pmatrix} 2\Phi \\ \hline \boxed{2\alpha I_n} \end{pmatrix}$$

For any $\mathbf{z} \in \mathbb{Z}^n$,
$\|2\alpha I_n \mathbf{z} - \boldsymbol{\alpha}\|_p^p = \alpha^p n$
if and only if $\mathbf{z} \in \{0,1\}^n$.

Otherwise, $\|2\alpha I_n \mathbf{z} - \boldsymbol{\alpha}\|_p^p \geq \alpha^p(n+2)$.

$$\begin{array}{rc} & t_1 \\ & t_2 \\ & \vdots \\ & t_m \\ \mathbf{t} := & \\ \hline & \alpha \\ & \alpha \\ \boldsymbol{\alpha} := & \vdots \\ & \alpha \end{array}$$

# 2-SAT
# (a very special case…)

$\Phi \in \mathbb{R}^{m \times n}$

$$\mathbf{B} := \left(\begin{array}{c} 2\Phi \\ \hline \boxed{2\alpha I_n} \end{array}\right)$$

For any $\mathbf{z} \in \mathbb{Z}^n$,
$\|2\alpha I_n \mathbf{z} - \boldsymbol{\alpha}\|_p^p = \alpha^p n$
if and only if $\mathbf{z} \in \{0,1\}^n$.

Otherwise, $\|2\alpha I_n \mathbf{z} - \boldsymbol{\alpha}\|_p^p \geq \alpha^p(n+2)$.

$$\begin{array}{rc} & t_1 \\ & t_2 \\ & \vdots \\ \mathbf{t} := & t_m \\ \hline & \boxed{\begin{array}{c} \alpha \\ \alpha \\ \vdots \\ \alpha \end{array}} \end{array}$$

$$\boldsymbol{\alpha} := \left(\begin{array}{c} \alpha \\ \alpha \\ \vdots \\ \alpha \end{array}\right)$$

We can assume that $\mathbf{z} \in \{0,1\}^n$!

# 2-SAT
# (a very special case…)

# 2-SAT
# (a very special case…)

$$\Phi_{i,j} = \begin{cases} 1 & x_j \text{ appears in clause i} \\ -1 & \overline{x}_j \text{ appears in clause i} \\ 0 & \text{otherwise} \end{cases}$$

# 2-SAT
# (a very special case…)

$$\Phi_{i,j} = \begin{cases} 1 & x_j \text{ appears in clause i} \\ -1 & \overline{x}_j \text{ appears in clause i} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{z} \in \{0,1\}^n$$

# 2-SAT
# (a very special case…)

$$\Phi_{i,j} = \begin{cases} 1 & x_j \text{ appears in clause i} \\ -1 & \overline{x}_j \text{ appears in clause i} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{z} \in \{0,1\}^n$$

$$\Phi_i \mathbf{z} = (\#\text{ positive literals satisfied}) - (\#\text{ of negated literals } \textit{not} \text{ satisfied})$$

# 2-SAT
# (a very special case…)

$$\Phi_{i,j} = \begin{cases} 1 & x_j \text{ appears in clause i} \\ -1 & \overline{x}_j \text{ appears in clause i} \\ 0 & \text{otherwise} \end{cases}$$

$$t_i := 3 - 2 \cdot (\#\text{ of negated literals in clause } i)$$

$$\mathbf{z} \in \{0,1\}^n$$

$$\Phi_i \mathbf{z} = (\#\text{ positive literals satisfied}) - (\#\text{ of negated literals } \textit{not} \text{ satisfied})$$

# 2-SAT
# (a very special case…)

$$\Phi_{i,j} = \begin{cases} 1 & x_j \text{ appears in clause i} \\ -1 & \overline{x}_j \text{ appears in clause i} \\ 0 & \text{otherwise} \end{cases}$$

$$t_i := 3 - 2 \cdot (\# \text{ of negated literals in clause } i)$$

$$\mathbf{z} \in \{0,1\}^n$$

$$\Phi_i \mathbf{z} = (\# \text{ positive literals satisfied}) - (\# \text{ of negated literals } \textit{not} \text{ satisfied})$$

$$2\Phi_i \mathbf{z} - t_i = 2(\# \text{ satisfied literals}) - 3$$

# 2-SAT
# (a very special case…)

$$\Phi_{i,j} = \begin{cases} 1 & x_j \text{ appears in clause i} \\ -1 & \overline{x}_j \text{ appears in clause i} \\ 0 & \text{otherwise} \end{cases}$$

$$t_i := 3 - 2 \cdot (\# \text{ of negated literals in clause } i)$$

$$\mathbf{z} \in \{0,1\}^n$$

$$\Phi_i \mathbf{z} = (\# \text{ positive literals satisfied}) - (\# \text{ of negated literals } \textit{not} \text{ satisfied})$$

$$2\Phi_i \mathbf{z} - t_i = 2(\# \text{ satisfied literals}) - 3$$

$$|2\Phi_i \mathbf{z} - t_i| = \begin{cases} 1 & i\text{th clause is satisfied} \\ 3 & \text{otherwise} \end{cases}$$

# 2-SAT
# (a very special case…)

$$\Phi_{i,j} = \begin{cases} 1 & x_j \text{ appears in clause i} \\ -1 & \overline{x}_j \text{ appears in clause i} \\ 0 & \text{otherwise} \end{cases}$$

$$t_i := 3 - 2 \cdot (\text{\# of negated literals in clause } i)$$

$$\mathbf{z} \in \{0,1\}^n$$

$$\Phi_i \mathbf{z} = (\text{\# positive literals satisfied}) - (\text{\# of negated literals } \textit{not} \text{ satisfied})$$

$$2\Phi_i \mathbf{z} - t_i = 2(\text{\# satisfied literals}) - 3$$

$$|2\Phi_i \mathbf{z} - t_i| = \begin{cases} 1 & i\text{th clause is satisfied} \\ 3 & \text{otherwise} \end{cases}$$

$$\begin{aligned} \|\Phi \mathbf{z} - \mathbf{t}\|_p^p &= (\text{\# satisfied clauses}) + 3^p(\text{\# unsatisfied clauses}) \\ &= 3^p m - (3^p - 1)(\text{\# satisfied clauses}) \end{aligned}$$

# 2-SAT
# (a very special case…)

$$\Phi_{i,j} = \begin{cases} 1 & x_j \text{ a} \\ -1 & \overline{x}_j \text{ a} \\ 0 & \text{oth} \end{cases} \qquad \text{n clause } i)$$

$\mathrm{dist}_p(\mathbf{t}, \mathcal{L})$ tells us exactly the maximal number of satisfied clauses. So, this reduction works for Max-2-SAT.

Max-2-SAT is hard!

$$\Phi_i \mathbf{z} = (\# \text{ po} \qquad \text{tisfied})$$

$$2\Phi_i \mathbf{z} - t_i = 2(\# \text{ satisfied literals}) - 3$$

$$|2\Phi_i \mathbf{z} - t_i| = \begin{cases} 1 & i\text{th clause is satisfied} \\ 3 & \text{otherwise} \end{cases}$$

$$\begin{aligned} \|\Phi \mathbf{z} - \mathbf{t}\|_p^p &= (\# \text{ satisfied clauses}) + 3^p(\# \text{ unsatisfied clauses}) \\ &= 3^p m - (3^p - 1)(\# \text{ satisfied clauses}) \end{aligned}$$

# 2-SAT
# (a very special case…)

$$\Phi_{i,j} = \begin{cases} 1 & x_j \text{ a} \\ -1 & \overline{x}_j \text{ a} \\ 0 & \text{oth} \end{cases}$$ clause $i$)

$\mathrm{dist}_p(\mathbf{t}, \mathcal{L})$ tells us exactly the maximal number of satisfied clauses. So, this reduction works for Max-2-SAT.

Max-2-SAT is hard!

$\Phi_i \mathbf{z} = (\# \text{ po}$ tisfied)

$2\Phi_i$ $- 3$

$|2\Phi_i$ fied

$\|\Phi \mathbf{z} - \mathbf{t}\|_p^p = (\#$ satisfied clauses)

$= 3^p m$ ses)

# What did we just prove?

# What did we just prove?

- Max-2-SAT is “ETH-hard.” I.e., assuming SETH (or just ETH), there is no $2^{o(n)}$-time algorithm for Max-2-SAT.

# What did we just prove?

- Max-2-SAT is “ETH-hard.” I.e., assuming SETH (or just ETH), there is no $2^{o(n)}$-time algorithm for Max-2-SAT.
    - Assuming SETH (or just ETH), there is no $2^{o(n)}$-time algorithm for CVP.

# What did we just prove?

- Max-2-SAT is "ETH-hard." I.e., assuming SETH (or just ETH), there is no $2^{o(n)}$-time algorithm for Max-2-SAT.
  - Assuming SETH (or just ETH), there is no $2^{o(n)}$-time algorithm for CVP.
  - (Result already known in folklore, and unpublished work [Yeom15].)

# What did we just prove?

- Max-2-SAT is “ETH-hard.” I.e., assuming SETH (or just ETH), there is no $2^{o(n)}$-time algorithm for Max-2-SAT.
  - Assuming SETH (or just ETH), there is no $2^{o(n)}$-time algorithm for CVP.
  - (Result already known in folklore, and unpublished work [Yeom15].)
- Fastest algorithm for Max-2-SAT runs in time $2^{\omega n/3} > 2^{0.78n}$ [Will05].

# What did we just prove?

- Max-2-SAT is “ETH-hard.” I.e., assuming SETH (or just ETH), there is no $2^{o(n)}$-time algorithm for Max-2-SAT.
  - Assuming SETH (or just ETH), there is no $2^{o(n)}$-time algorithm for CVP.
  - (Result already known in folklore, and unpublished work [Yeom15].)
- Fastest algorithm for Max-2-SAT runs in time $2^{\omega n/3} > 2^{0.78n}$ [Will05].
  - Assuming this is optimal, then there is no $2^{0.78n}$-time algorithm for $\mathsf{CVP}_p$ for any p.

# What did we just prove?

- Max-2-SAT is “ETH-hard.” I.e., assuming SETH (or just ETH), there is no $2^{o(n)}$-time algorithm for Max-2-SAT.
  - Assuming SETH (or just ETH), there is no $2^{o(n)}$-time algorithm for CVP.
  - (Result already known in folklore, and unpublished work [Yeom15].)
- Fastest algorithm for Max-2-SAT runs in time $2^{\omega n/3} > 2^{0.78n}$ [Will05].
  - Assuming this is optimal, then there is no $2^{0.78n}$-time algorithm for $\mathsf{CVP}_p$ for any p.
  - (Compare to $2^{n+o(n)}$-time algorithm for $\mathrm{CVP}_2$.)

# What did we just prove?

- Max-2-SAT is "ETH-hard." I.e., assuming SETH (or just ETH), there is no $2^{o(n)}$-time algorithm for Max-2-SAT.
  - Assuming SETH (or just ETH), there is no $2^{o(n)}$-time algorithm for CVP.
  - (Result already known in folklore, and unpublished work [Yeom15].)
- Fastest algorithm for Max-2-SAT runs in time $2^{\omega n/3} > 2^{0.78n}$ [Will05].
  - Assuming this is optimal, then there is no $2^{0.78n}$-time algorithm for $\mathsf{CVP}_p$ for any p.
  - (Compare to $2^{n+o(n)}$-time algorithm for $\mathrm{CVP}_2$.)

# What did we just prove?

- Max-2-SAT is "ETH-hard." I.e., assuming SETH (or just ETH), there is no $2^{o(n)}$-time algorithm for Max-2-SAT.
  - Assuming SETH (or just ETH), there is no $2^{o(n)}$-time algorithm for CVP.
  - (Result already known in folklore, and unpublished work [Yeom15].)
- Fastest algorithm for Max-2-SAT runs in time $2^{\omega n/3} > 2^{0.78n}$ [Will05].
  - Assuming this is optimal, then there is no $2^{0.78n}$-time algorithm for $\mathsf{CVP}_p$ for any p.
  - (Compare to $2^{n+o(n)}$-time algorithm for $\mathrm{CVP}_2$.)

Not a very safe assumption…

# Generalization to k-SAT?

# Generalization to k-SAT?

$$2\Phi_i \mathbf{z} - t_i = 2(\#\text{ satisfied literals}) - 3$$

# Generalization to k-SAT?

$$2\Phi_i \mathbf{z} - t_i = 2(\#\ \text{satisfied literals}) - 3$$

- A 2-SAT clause is satisfied if and only if the number of satisfied literals is 1 or 2.

# Generalization to k-SAT?

$$2\Phi_i\mathbf{z} - t_i = 2(\#\text{ satisfied literals}) - 3$$

- A 2-SAT clause is satisfied if and only if the number of satisfied literals is 1 or 2.
- Therefore $|2\Phi_i\mathbf{z} - t_i| = 1$ if and only if the clause is satisfied.

# Generalization to k-SAT?

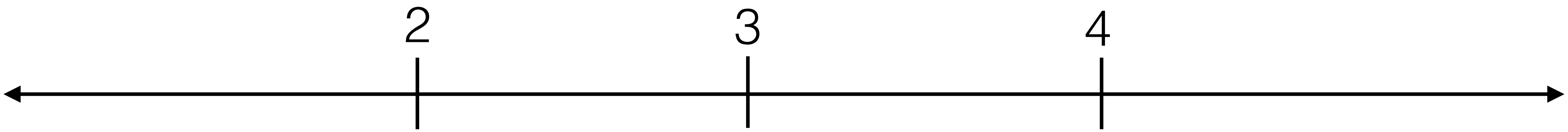
$$2\Phi_i \mathbf{z} - t_i = 2(\#\ \text{satisfied literals}) - 3$$

- A 2-SAT clause is satisfied if and only if the number of satisfied literals is 1 or 2.
- Therefore $|2\Phi_i \mathbf{z} - t_i| = 1$ if and only if the clause is satisfied.
  - 2 and 4 are equidistant from 3!

# Generalization to k-SAT?

$$2\Phi_i\mathbf{z} - t_i = 2(\#\text{ satisfied literals}) - 3$$

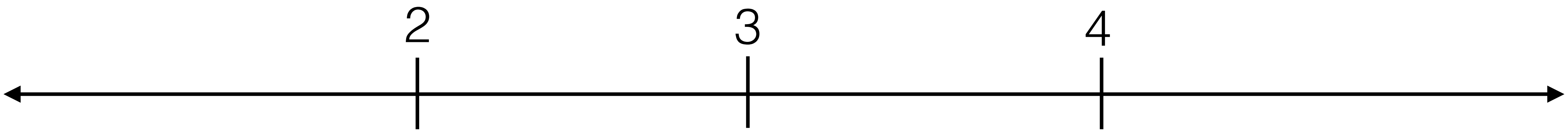
- A 2-SAT clause is satisfied if and only if the number of satisfied literals is 1 or 2.
- Therefore $|2\Phi_i\mathbf{z} - t_i| = 1$ if and only if the clause is satisfied.
  - 2 and 4 are equidistant from 3!



# Generalization to k-SAT?

$$2\Phi_i \mathbf{z} - t_i = 2(\#\ \text{satisfied literals}) - 3$$

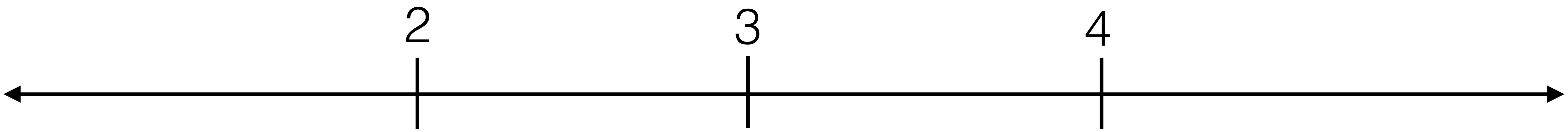
- A 2-SAT clause is satisfied if and only if the number of satisfied literals is 1 or 2.
- Therefore $|2\Phi_i \mathbf{z} - t_i| = 1$ if and only if the clause is satisfied.
    - 2 and 4 are equidistant from 3!
- A 3-SAT clause is satisfied if and only if the number of satisfied literals is 1, 2, or 3.



# Generalization to k-SAT?

$$2\Phi_i \mathbf{z} - t_i = 2(\#\text{ satisfied literals}) - 3$$

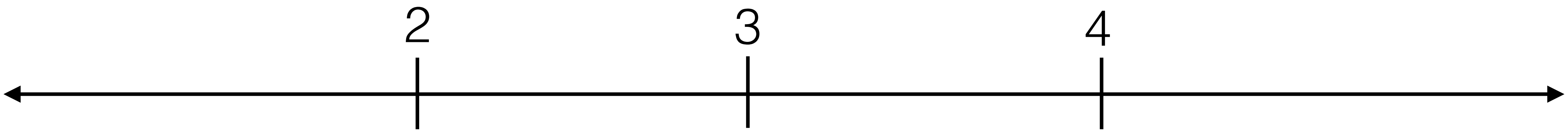
- A 2-SAT clause is satisfied if and only if the number of satisfied literals is 1 or 2.
- Therefore $|2\Phi_i \mathbf{z} - t_i| = 1$ if and only if the clause is satisfied.
    - 2 and 4 are equidistant from 3!
- A 3-SAT clause is satisfied if and only if the number of satisfied literals is 1, 2, or 3.
    - If we try the same construction with 3-SAT, we'll fail.



# Generalization to k-SAT?

$$2\Phi_i \mathbf{z} - t_i = 2(\#\text{ satisfied literals}) - 3$$

- A 2-SAT clause is satisfied if and only if the number of satisfied literals is 1 or 2.
- Therefore $|2\Phi_i \mathbf{z} - t_i| = 1$ if and only if the clause is satisfied.
    - 2 and 4 are equidistant from 3!
- A 3-SAT clause is satisfied if and only if the number of satisfied literals is 1, 2, or 3.
    - If we try the same construction with 3-SAT, we'll fail.
    - We can't find 3 distinct numbers that are equidistant from some other number…



# Generalization to k-SAT?

# Generalization to k-SAT?

We can’t find many distinct numbers that are equidistant from some other number…

# Generalization to k-SAT?

We can’t find many distinct numbers that are equidistant from some other number…

But, we can find many distinct *vectors* that are equidistant from some other *vector*

# Generalization to k-SAT?

We can’t find many distinct numbers that are equidistant from some other number…

But, we can find many distinct *vectors* that are equidistant from some other *vector*

$$\Phi_{i,j} = \begin{cases} 1 & x_j \text{ appears in clause i} \\ -1 & \overline{x}_j \text{ appears in clause i} \\ 0 & \text{otherwise} \end{cases}$$

# Generalization to k-SAT?

We can’t find many distinct numbers that are equidistant from some other number…

But, we can find many distinct *vectors* that are equidistant from some other *vector*

$$\Phi_{i,j} = \xcancel{\begin{cases} 1 & x_j \text{ appears in clause i} \\ -1 & \overline{x}_j \text{ appears in clause i} \\ 0 & \text{otherwise} \end{cases}}$$

# Generalization to k-SAT?

We can't find many distinct numbers that are equidistant from some other number…

But, we can find many distinct *vectors* that are equidistant from some other *vector*

$$\Phi_{i,j} = \begin{cases} 1 & \xcancel{x_j \text{ appears in clause i}} \\ -1 & \xcancel{\overline{x}_j \text{ appears in clause i}} \\ 0 & \xcancel{\text{otherwise}} \end{cases}$$

$$\Phi_{i,j} := \begin{cases} \mathbf{v}_\ell & \text{if } x_j \text{ is the } \ell\text{th literal in clause } i \\ -\mathbf{v}_\ell & \text{if } \overline{x}_j \text{ is the } \ell\text{th literal in clause } i \\ \mathbf{0} & \text{otherwise} \end{cases}$$

# Generalization to k-SAT?

We can’t find many distinct numbers that are equidistant from some other number…

But, we can find many distinct *vectors* that are equidistant from some other *vector*

$$\Phi_{i,j} = \xcancel{\begin{cases} 1 & x_j \text{ appears in clause i} \\ -1 & \overline{x}_j \text{ appears in clause i} \\ 0 & \text{otherwise} \end{cases}}$$

$$\Phi_{i,j} := \begin{cases} \mathbf{v}_\ell & \text{if } x_j \text{ is the } \ell\text{th literal in clause } i \\ -\mathbf{v}_\ell & \text{if } \overline{x}_j \text{ is the } \ell\text{th literal in clause } i \\ 0 & \text{otherwise} \end{cases}$$

$$t_i := 3 - 2 \cdot (\#\text{ of negated literals in clause } i)$$

# Generalization to k-SAT?

We can't find many distinct numbers that are equidistant from some other number…

But, we can find many distinct *vectors* that are equidistant from some other *vector*

$$\xcancel{\Phi_{i,j} = \begin{cases} 1 & x_j \text{ appears in clause i} \\ -1 & \overline{x}_j \text{ appears in clause i} \\ 0 & \text{otherwise} \end{cases}}$$

$$\Phi_{i,j} := \begin{cases} \mathbf{v}_\ell & \text{if } x_j \text{ is the } \ell\text{th literal in clause } i \\ -\mathbf{v}_\ell & \text{if } \overline{x}_j \text{ is the } \ell\text{th literal in clause } i \\ \mathbf{0} & \text{otherwise} \end{cases}$$

$$\xcancel{t_i := 3 - 2 \cdot (\#\text{ of negated literals in clause } i)}$$

# Generalization to k-SAT?

We can’t find many distinct numbers that are equidistant from some other number…

But, we can find many distinct *vectors* that are equidistant from some other *vector*

$$\Phi_{i,j} = \xcancel{\begin{cases} 1 & x_j \text{ appears in clause i} \\ -1 & \overline{x}_j \text{ appears in clause i} \\ 0 & \text{otherwise} \end{cases}}$$

$$\Phi_{i,j} := \begin{cases} \mathbf{v}_\ell & \text{if } x_j \text{ is the } \ell\text{th literal in clause } i \\ -\mathbf{v}_\ell & \text{if } \overline{x}_j \text{ is the } \ell\text{th literal in clause } i \\ \mathbf{0} & \text{otherwise} \end{cases}$$

$$\xcancel{t_i := 3 - 2 \cdot (\#\text{ of negated literals in clause } i)}$$

$$\mathbf{t}_i := \mathbf{t}^* - \sum_{\ell\text{th literal negated}} \mathbf{v}_\ell$$

# Generalization to k-SAT?

We can't find many distinct numbers that are equidistant from some other number…

But, we can find many distinct *vectors* that are equidistant from some other *vector*

$$\xcancel{\Phi_{i,j} = \begin{cases} 1 & x_j \text{ appears in clause i} \\ -1 & \overline{x}_j \text{ appears in clause i} \\ 0 & \text{otherwise} \end{cases}}$$

$$\Phi_{i,j} := \begin{cases} \mathbf{v}_\ell & \text{if } x_j \text{ is the } \ell\text{th literal in clause } i \\ -\mathbf{v}_\ell & \text{if } \overline{x}_j \text{ is the } \ell\text{th literal in clause } i \\ 0 & \text{otherwise} \end{cases}$$

$$\xcancel{t_i := 3 - 2 \cdot (\#\text{ of negated literals in clause } i)}$$

$$\mathbf{t}_i := \mathbf{t}^* - \sum_{\ell\text{th literal negated}} \mathbf{v}_\ell$$

$$\text{For } \mathbf{z} \in \{0,1\}^n,\ \Phi_i \mathbf{z} - \mathbf{t}_i = \sum_{\mathbf{z} \text{ satisfies } \ell\text{th literal}} \mathbf{v}_\ell - \mathbf{t}^*$$

# Generalization to k-SAT

# Generalization to k-SAT

$$\text{For } \mathbf{z} \in \{0,1\}^n,\ \Phi_i \mathbf{z} - \mathbf{t}_i = \sum_{\mathbf{z} \text{ satisfies } \ell\text{th literal}} \mathbf{v}_\ell - \mathbf{t}^*$$

# Generalization to k-SAT

$$\text{For } \mathbf{z} \in \{0,1\}^n,\ \Phi_i \mathbf{z} - \mathbf{t}_i = \sum_{\mathbf{z} \text{ satisfies } \ell\text{th literal}} \mathbf{v}_\ell - \mathbf{t}^*$$

We want the norm of this vector to be *constant* and smaller than the norm of $\mathbf{t}^*$ whenever the sum is non-empty.

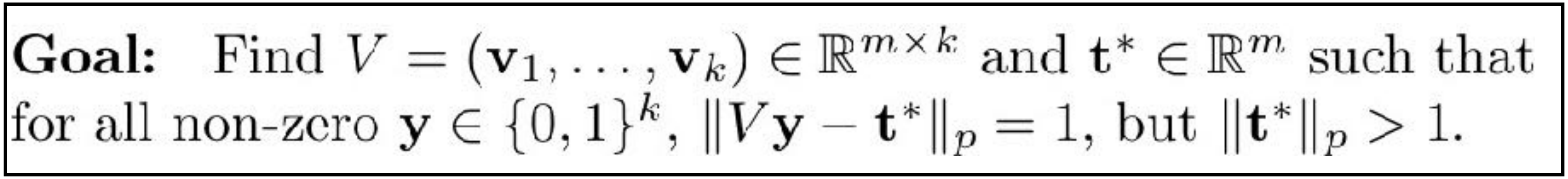
# Generalization to k-SAT

$$\boxed{\text{For } \mathbf{z} \in \{0,1\}^n,\ \Phi_i \mathbf{z} - \mathbf{t}_i = \sum_{\mathbf{z} \text{ satisfies } \ell\text{th literal}} \mathbf{v}_\ell - \mathbf{t}^*}$$

We want the norm of this vector to be *constant* and smaller than the norm of $\mathbf{t}^*$ whenever the sum is non-empty.

**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

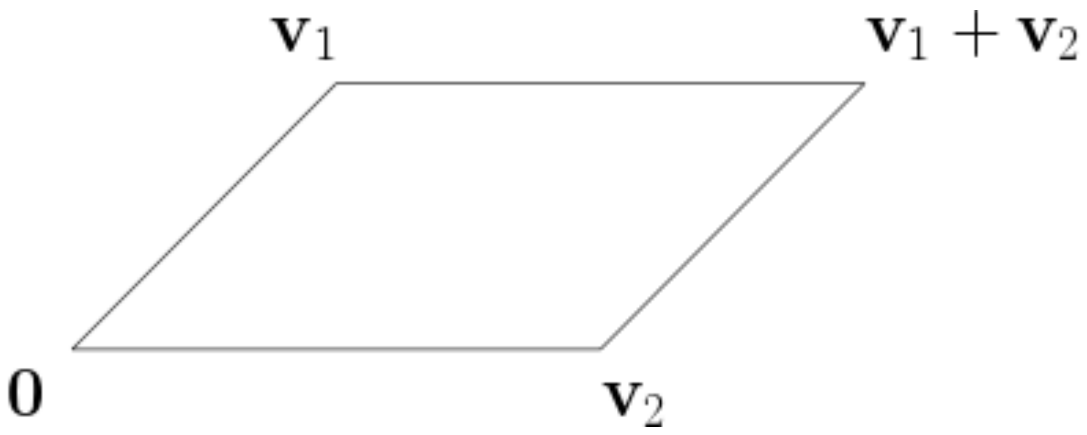
# Isolating Parallelepipeds



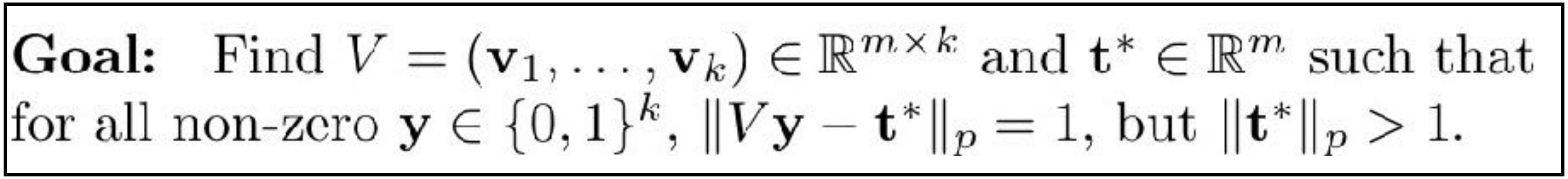
**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

# Isolating Parallelepipeds

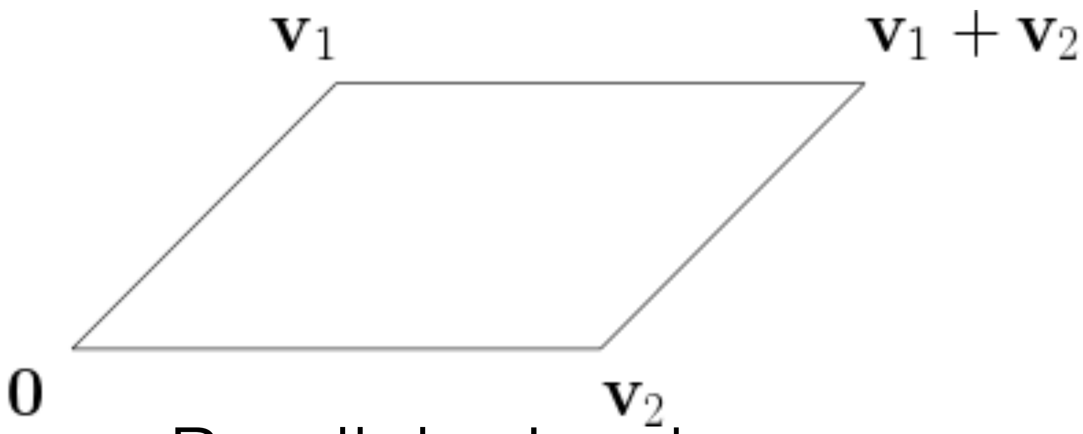
**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.



# Isolating Parallelepipeds



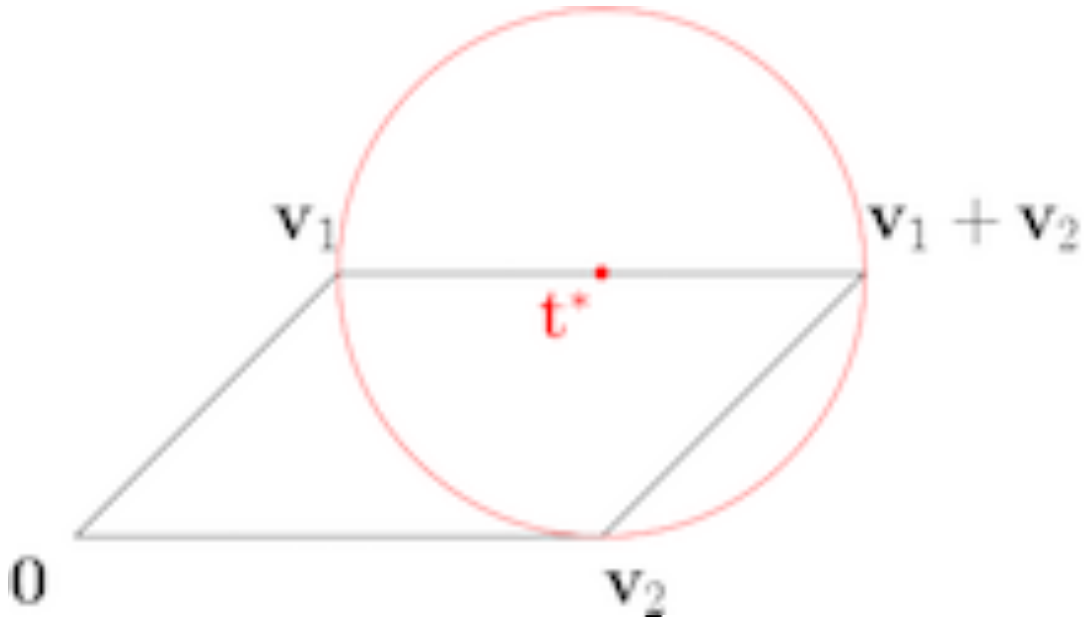
**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.



Parallelepiped

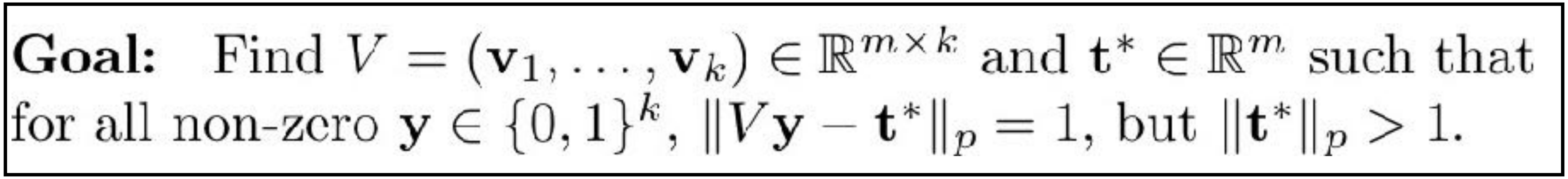
# Isolating Parallelepipeds

**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

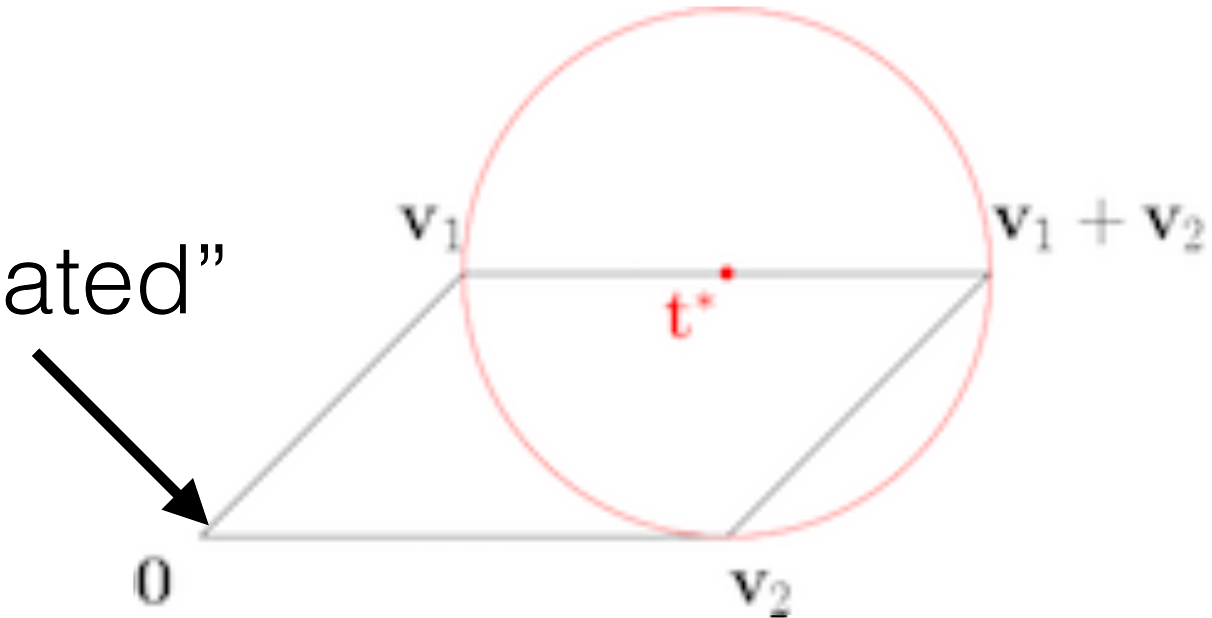


Parallelepiped

# Isolating Parallelepipeds



**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

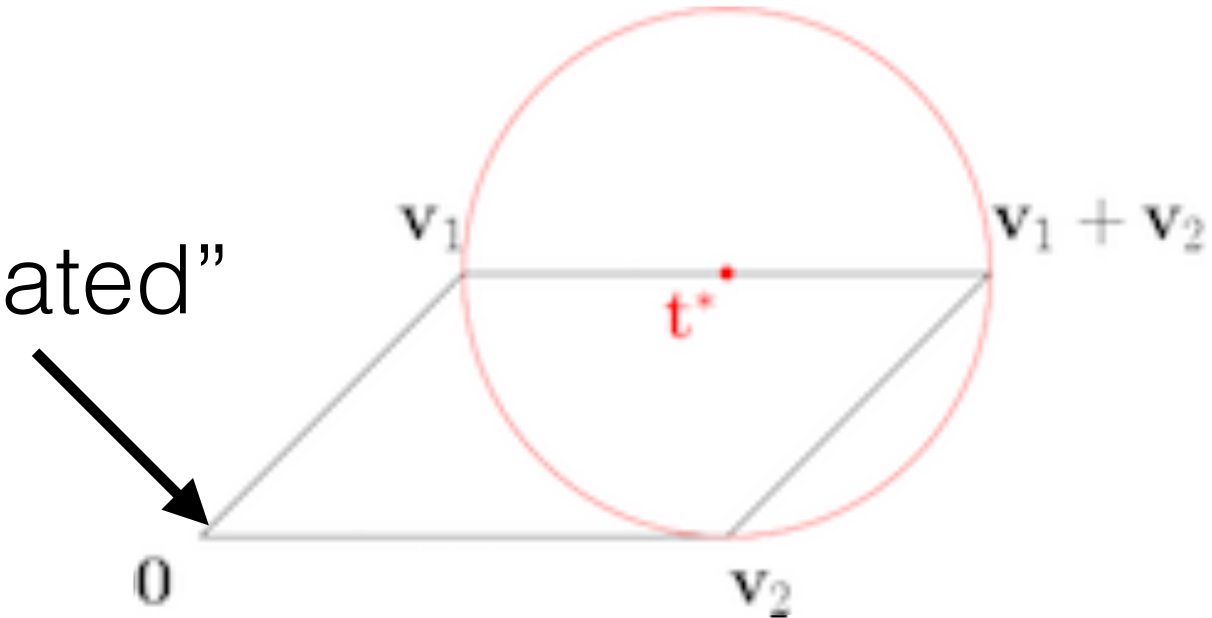


“Isolated”

Parallelepiped

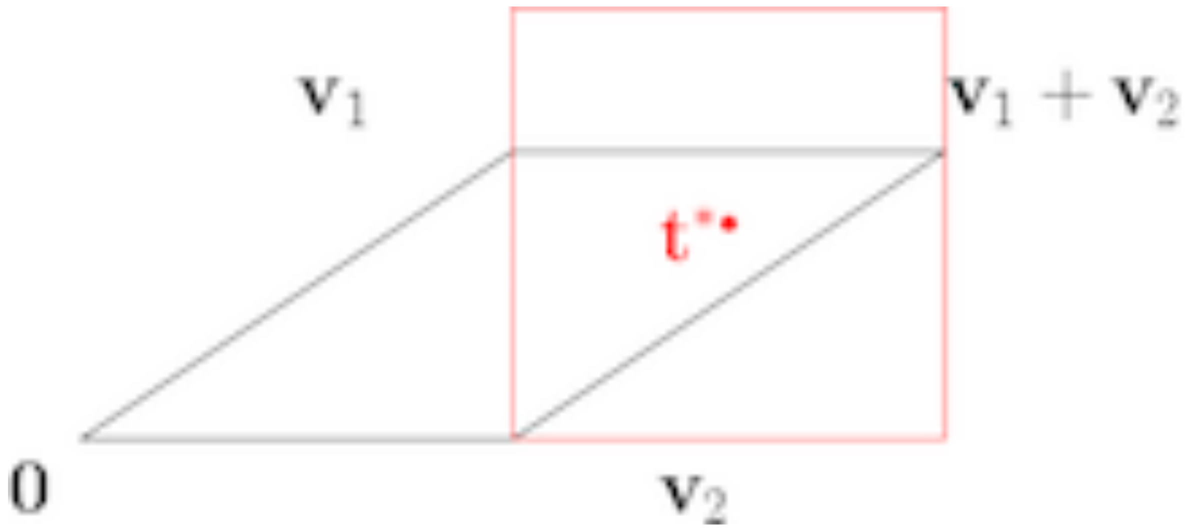
# Isolating Parallelepipeds

**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

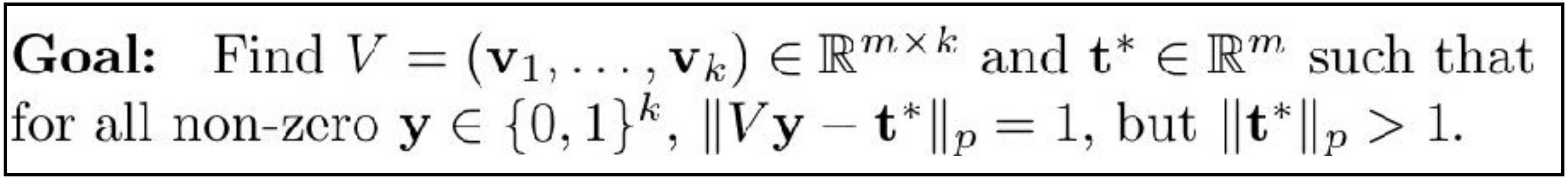


"Isolated"

Parallelepiped

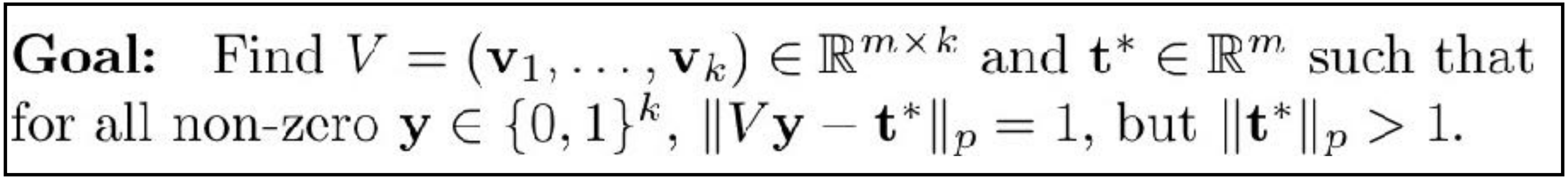


# Isolating Parallelepipeds



**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0, 1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

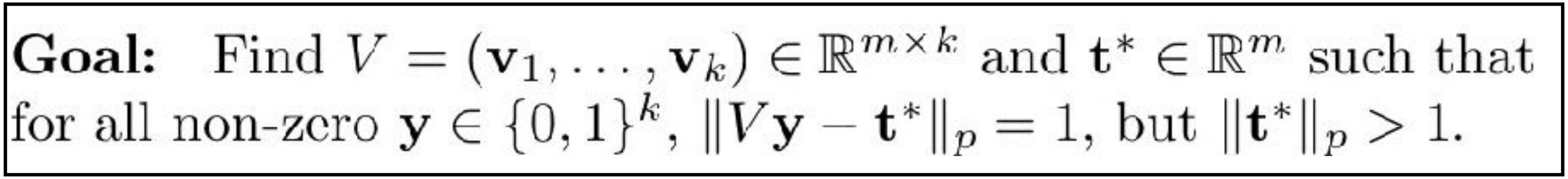
# Isolating Parallelepipeds



**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

Do these things even exist for large k?

# Isolating Parallelepipeds

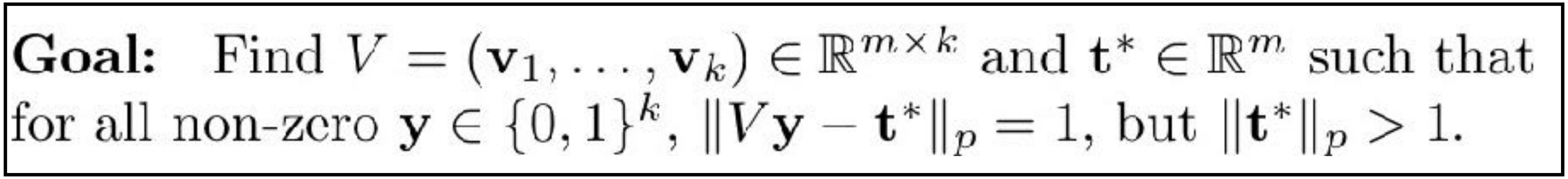


**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

Do these things even exist for large k?

- "Most of the time," they do!

# Isolating Parallelepipeds

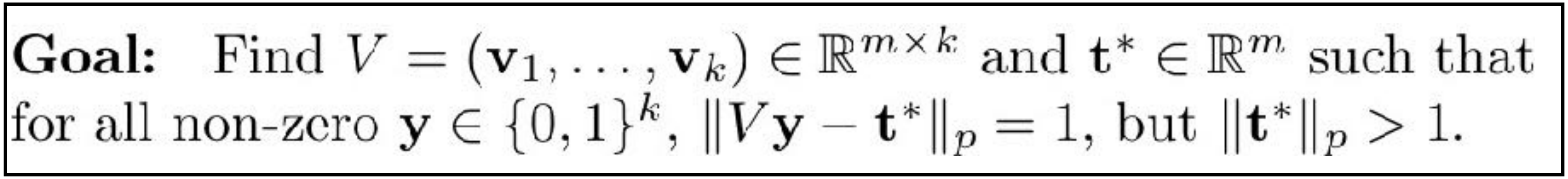


**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

Do these things even exist for large k?

- "Most of the time," they do!
- For odd integers p, they exist for all k.

# Isolating Parallelepipeds

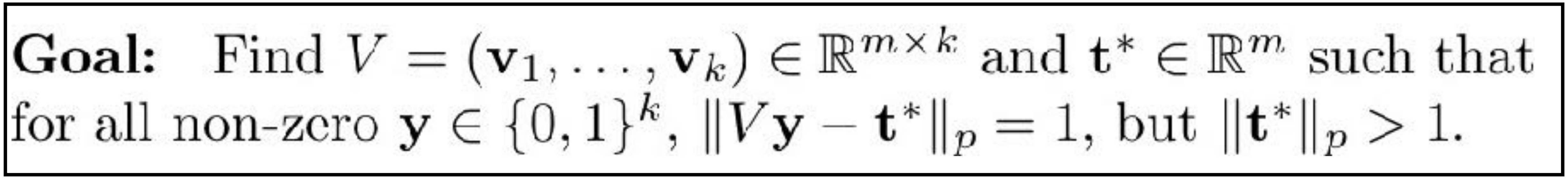


**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

Do these things even exist for large k?

- "Most of the time," they do!
- For odd integers p, they exist for all k.
    - If SETH holds, then for all odd integers p, there is no $2^{0.99n}$-time algorithm for $\mathsf{CVP}_p$!!!

# Isolating Parallelepipeds

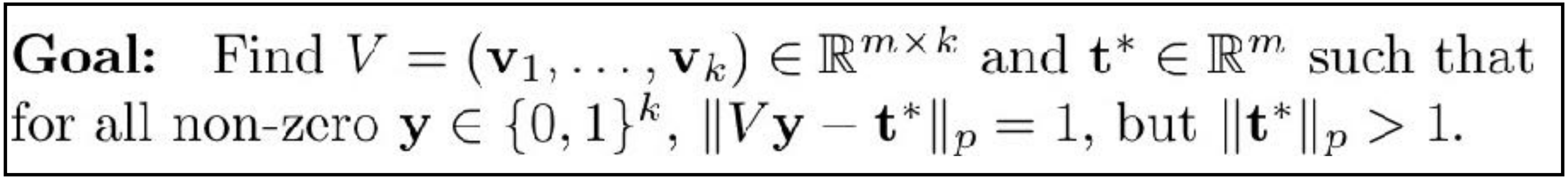


**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

Do these things even exist for large k?

- "Most of the time," they do!
- For odd integers p, they exist for all k.
  - If SETH holds, then for all odd integers p, there is no $2^{0.99n}$-time algorithm for $\mathsf{CVP}_p$!!!
  - (Try p = 1 yourself.)

# Isolating Parallelepipeds

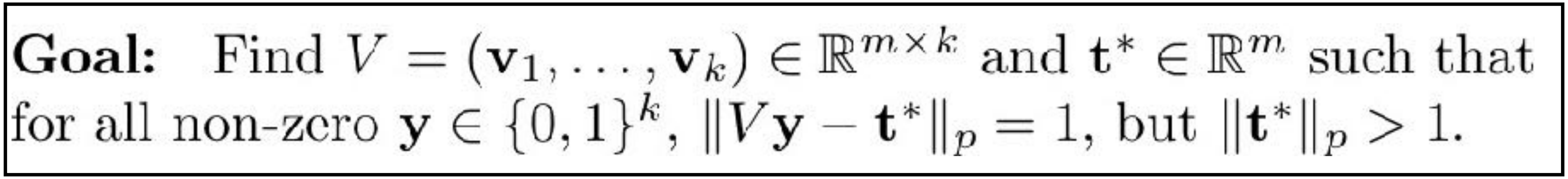


**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

Do these things even exist for large k?

- "Most of the time," they do!
- For odd integers p, they exist for all k.
  - If SETH holds, then for all odd integers p, there is no $2^{0.99n}$-time algorithm for $\mathsf{CVP}_p$!!!
  - (Try p = 1 yourself.)
- For even integers p, they exist if and only if $k \leq p$.

# Isolating Parallelepipeds

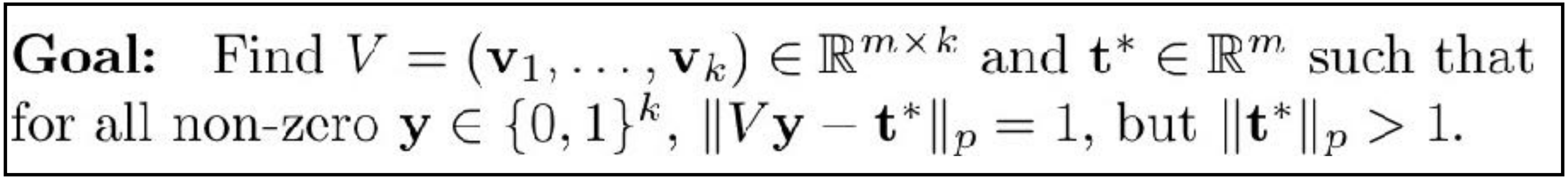


**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

Do these things even exist for large k?

- "Most of the time," they do!
- For odd integers p, they exist for all k.
    - If SETH holds, then for all odd integers p, there is no $2^{0.99n}$-time algorithm for $\mathsf{CVP}_p$!!!
    - (Try p = 1 yourself.)
- For even integers p, they exist if and only if $k \leq p$.
    - Two is an even integer :(.

# Isolating Parallelepipeds

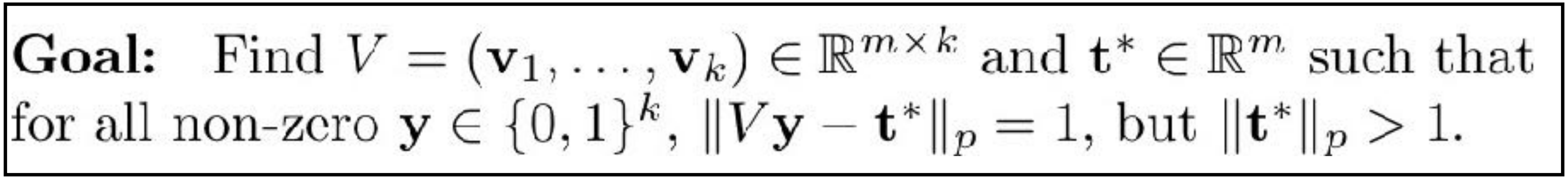


**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

Do these things even exist for large k?

- "Most of the time," they do!
- For odd integers p, they exist for all k.
  - If SETH holds, then for all odd integers p, there is no $2^{0.99n}$-time algorithm for $\mathsf{CVP}_p$!!!
  - (Try p = 1 yourself.)
- For even integers p, they exist if and only if $k \leq p$.
  - Two is an even integer :(.
- For any fixed k, they exist for all but finitely many values of p.

# Isolating Parallelepipeds

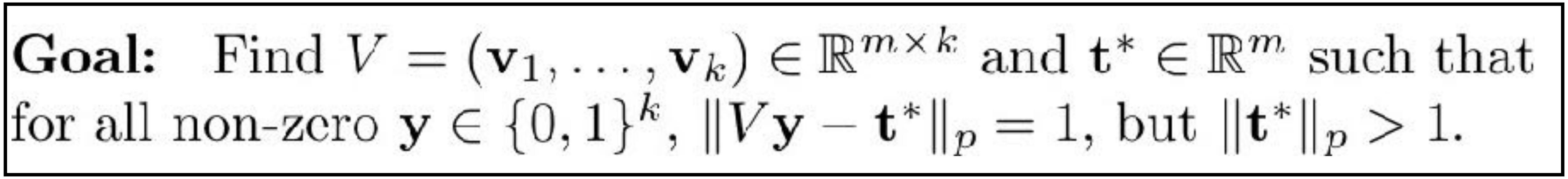


**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

Do these things even exist for large k?

- "Most of the time," they do!
- For odd integers p, they exist for all k.
    - If SETH holds, then for all odd integers p, there is no $2^{0.99n}$-time algorithm for $\mathsf{CVP}_p$!!!
    - (Try p = 1 yourself.)
- For even integers p, they exist if and only if $k \leq p$.
    - Two is an even integer :(.
- For any fixed k, they exist for all but finitely many values of p.
- For any k and any $p(n) = p_0 + \delta(n)$ with $\delta(n) \neq 0$ and $\delta(n) \to 0$, they exist for sufficiently large n.

# Isolating Parallelepipeds

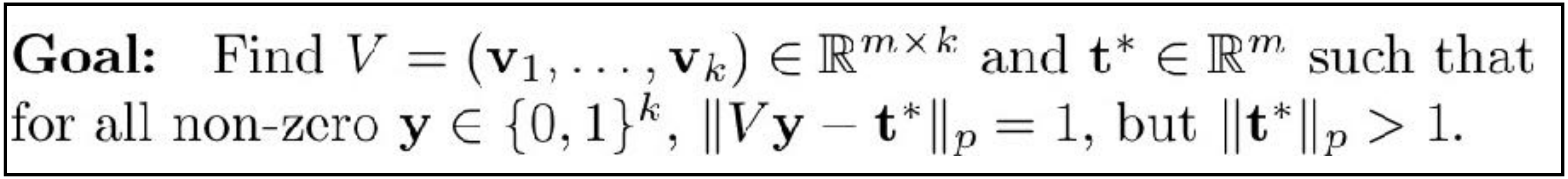


**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

Do these things even exist for large k?

- "Most of the time," they do!
- For odd integers p, they exist for all k.
  - If SETH holds, then for all odd integers p, there is no $2^{0.99n}$-time algorithm for $\mathsf{CVP}_p$!!!
  - (Try p = 1 yourself.)
- For even integers p, they exist if and only if $k \leq p$.
  - Two is an even integer :(.
- For any fixed k, they exist for all but finitely many values of p.
- For any k and any $p(n) = p_0 + \delta(n)$ with $\delta(n) \neq 0$ and $\delta(n) \to 0$, they exist for sufficiently large n.
  - If SETH holds, then no $2^{0.99n}$ -time algorithm solves $\mathsf{CVP}_p$ for such p!!!

# Isolating Parallelepipeds



**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

# Isolating Parallelepipeds

**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

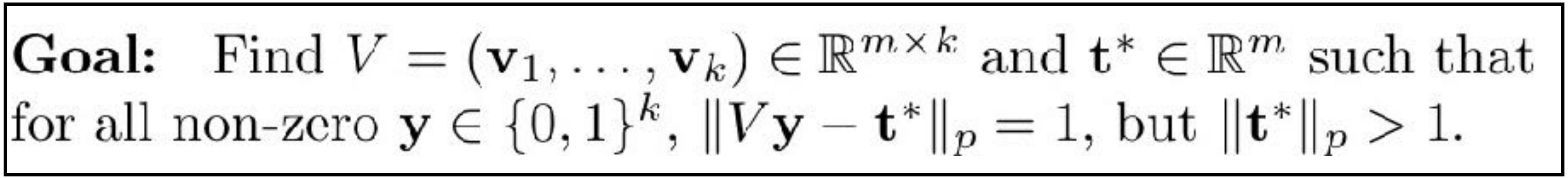
$$V = \begin{pmatrix} \\ \\ \\ \end{pmatrix}$$

# Isolating Parallelepipeds

**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

$$V = \begin{pmatrix} 1 & 1 & 1 \\ & & \\ & & \\ & & \\ & & \end{pmatrix}$$

# Isolating Parallelepipeds



**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

$$V = \begin{array}{c} \left(\begin{array}{ccc} 1 & 1 & 1 \\ & & \\ & & \\ & & \\ & & \end{array}\right) \end{array} \begin{array}{l} y_1 + y_2 + y_3 \\ \\ \\ \\ \\ \end{array}$$

# Isolating Parallelepipeds

**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

$$V = \left(\begin{array}{rrr} 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -1 & 1 \\ -1 & 1 & 1 \end{array}\right) \begin{array}{l} y_1 + y_2 + y_3 \\ \\ \\ \\ \end{array}$$

# Isolating Parallelepipeds

**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

$$V = \left(\begin{array}{rrr} 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -1 & 1 \\ -1 & 1 & 1 \\ & & \\ & & \\ & & \end{array}\right) \begin{array}{r} y_1 + y_2 + y_3 \\ y_1 + y_2 - y_3 \\ y_1 - y_2 + y_3 \\ -y_1 + y_2 + y_3 \\ \\ \\ \\ \end{array}$$

# Isolating Parallelepipeds

**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

$$V = \begin{array}{c} \alpha_0 \\ \alpha_1 \\ \alpha_1 \\ \alpha_1 \\ \\ \\ \\ \end{array} \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -1 & 1 \\ -1 & 1 & 1 \\ & & \\ & & \\ & & \end{pmatrix} \begin{array}{r} y_1 + y_2 + y_3 \\ y_1 + y_2 - y_3 \\ y_1 - y_2 + y_3 \\ -y_1 + y_2 + y_3 \\ \\ \\ \\ \end{array}$$

# Isolating Parallelepipeds

**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

$$V = \begin{array}{c} \alpha_0 \\ \alpha_1 \\ \alpha_1 \\ \alpha_1 \\ \\ \\ \\ \end{array} \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -1 & 1 \\ -1 & 1 & 1 \\ 1 & -1 & -1 \\ -1 & 1 & -1 \\ -1 & -1 & 1 \end{pmatrix} \begin{array}{l} y_1 + y_2 + y_3 \\ y_1 + y_2 - y_3 \\ y_1 - y_2 + y_3 \\ -y_1 + y_2 + y_3 \\ \\ \\ \\ \end{array}$$

# Isolating Parallelepipeds

**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

$$V = \begin{array}{c} \alpha_0 \\ \alpha_1 \\ \alpha_1 \\ \alpha_1 \\ \\ \\ \\ \end{array} \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -1 & 1 \\ -1 & 1 & 1 \\ 1 & -1 & -1 \\ -1 & 1 & -1 \\ -1 & -1 & 1 \end{pmatrix} \begin{array}{l} y_1 + y_2 + y_3 \\ y_1 + y_2 - y_3 \\ y_1 - y_2 + y_3 \\ -y_1 + y_2 + y_3 \\ y_1 - y_2 - y_3 \\ -y_1 + y_2 - y_3 \\ -y_1 - y_2 + y_3 \end{array}$$

# Isolating Parallelepipeds

**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

$$V = \begin{array}{c} \alpha_0 \\ \alpha_1 \\ \alpha_1 \\ \alpha_1 \\ \\ \\ \\ \\ \end{array} \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -1 & 1 \\ -1 & 1 & 1 \\ 1 & -1 & -1 \\ -1 & 1 & -1 \\ -1 & -1 & 1 \\ -1 & -1 & -1 \end{pmatrix} \begin{array}{l} y_1 + y_2 + y_3 \\ y_1 + y_2 - y_3 \\ y_1 - y_2 + y_3 \\ -y_1 + y_2 + y_3 \\ y_1 - y_2 - y_3 \\ -y_1 + y_2 - y_3 \\ -y_1 - y_2 + y_3 \\ \\ \end{array}$$

# Isolating Parallelepipeds

**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

$$V = \begin{array}{r} \alpha_0 \\ \alpha_1 \\ \alpha_1 \\ \alpha_1 \\ \\ \\ \\ \\ \end{array} \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -1 & 1 \\ -1 & 1 & 1 \\ 1 & -1 & -1 \\ -1 & 1 & -1 \\ -1 & -1 & 1 \\ -1 & -1 & -1 \end{pmatrix} \begin{array}{r} y_1 + y_2 + y_3 \\ y_1 + y_2 - y_3 \\ y_1 - y_2 + y_3 \\ -y_1 + y_2 + y_3 \\ y_1 - y_2 - y_3 \\ -y_1 + y_2 - y_3 \\ -y_1 - y_2 + y_3 \\ -y_1 - y_2 - y_3 \end{array}$$

# Isolating Parallelepipeds

**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

$$V = \begin{array}{c} \alpha_0 \\ \alpha_1 \\ \alpha_1 \\ \alpha_1 \\ \alpha_2 \\ \alpha_2 \\ \alpha_2 \\ \alpha_3 \end{array} \left( \begin{array}{rrr} 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -1 & 1 \\ -1 & 1 & 1 \\ 1 & -1 & -1 \\ -1 & 1 & -1 \\ -1 & -1 & 1 \\ -1 & -1 & -1 \end{array} \right) \begin{array}{l} y_1 + y_2 + y_3 \\ y_1 + y_2 - y_3 \\ y_1 - y_2 + y_3 \\ -y_1 + y_2 + y_3 \\ y_1 - y_2 - y_3 \\ -y_1 + y_2 - y_3 \\ -y_1 - y_2 + y_3 \\ -y_1 - y_2 - y_3 \end{array}$$

# Isolating Parallelepipeds

**Goal:** Find $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{m \times k}$ and $\mathbf{t}^* \in \mathbb{R}^m$ such that for all non-zero $\mathbf{y} \in \{0,1\}^k$, $\|V\mathbf{y} - \mathbf{t}^*\|_p = 1$, but $\|\mathbf{t}^*\|_p > 1$.

$$V = \begin{array}{c} \alpha_0 \\ \alpha_1 \\ \alpha_1 \\ \alpha_1 \\ \alpha_2 \\ \alpha_2 \\ \alpha_2 \\ \alpha_3 \end{array} \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -1 & 1 \\ -1 & 1 & 1 \\ 1 & -1 & -1 \\ -1 & 1 & -1 \\ -1 & -1 & 1 \\ -1 & -1 & -1 \end{pmatrix} \begin{array}{c} y_1 + y_2 + y_3 \\ y_1 + y_2 - y_3 \\ y_1 - y_2 + y_3 \\ -y_1 + y_2 + y_3 \\ y_1 - y_2 - y_3 \\ -y_1 + y_2 - y_3 \\ -y_1 - y_2 + y_3 \\ -y_1 - y_2 - y_3 \end{array}$$

$$\mathbf{t}^* = \begin{pmatrix} \alpha_0 t^* \\ \alpha_1 t^* \\ \alpha_1 t^* \\ \alpha_1 t^* \\ \alpha_2 t^* \\ \alpha_2 t^* \\ \alpha_2 t^* \\ \alpha_3 t^* \end{pmatrix}$$

# Isolating Parallelepipeds

$$V = \begin{array}{r} \alpha_0 \\ \alpha_1 \\ \alpha_1 \\ \alpha_1 \\ \alpha_2 \\ \alpha_2 \\ \alpha_2 \\ \alpha_3 \end{array} \!\! \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -1 & 1 \\ -1 & 1 & 1 \\ 1 & -1 & -1 \\ -1 & 1 & -1 \\ -1 & -1 & 1 \\ -1 & -1 & -1 \end{pmatrix}$$

$$\mathbf{t}^* = \begin{pmatrix} \alpha_0 t^* \\ \alpha_1 t^* \\ \alpha_1 t^* \\ \alpha_1 t^* \\ \alpha_2 t^* \\ \alpha_2 t^* \\ \alpha_2 t^* \\ \alpha_3 t^* \end{pmatrix}$$

# Isolating Parallelepipeds

$$V = \begin{array}{c} \alpha_0 \\ \alpha_1 \\ \alpha_1 \\ \alpha_1 \\ \alpha_2 \\ \alpha_2 \\ \alpha_2 \\ \alpha_3 \end{array} \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -1 & 1 \\ -1 & 1 & 1 \\ 1 & -1 & -1 \\ -1 & 1 & -1 \\ -1 & -1 & 1 \\ -1 & -1 & -1 \end{pmatrix} \qquad \mathbf{t}^* = \begin{pmatrix} \alpha_0 t^* \\ \alpha_1 t^* \\ \alpha_1 t^* \\ \alpha_1 t^* \\ \alpha_2 t^* \\ \alpha_2 t^* \\ \alpha_2 t^* \\ \alpha_3 t^* \end{pmatrix}$$

$$\begin{aligned} \left\| V \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} - \mathbf{t}^* \right\|_p^p &= |\alpha_0|^p |2 - t^*|^p \\ &+ |\alpha_1|^p \cdot (|2 - t^*|^p + 2|t^*|^p) \\ &+ |\alpha_2|^p \cdot (|-2 - t^*|^p + 2|t^*|^p) \\ &+ |\alpha_3|^p \cdot |-2 - t^*|^p \end{aligned}$$

# Isolating Parallelepipeds

$$V = \begin{array}{c|ccc|} \alpha_0 & 1 & 1 & 1 \\ \alpha_2 & -1 & -1 & 1 \\ \alpha_3 & -1 & -1 & -1 \end{array} \qquad \begin{pmatrix} \alpha_0 t^* \\ \alpha_2 t^* \\ \alpha_3 t^* \end{pmatrix}$$

This is linear in the $|\alpha_i|^p$!

So, it suffices to find $t^*$ such that the resulting system of linear equations in the $|\alpha_i|^p$ has a (non-negative) solution.

$$\left\| V \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} - \mathbf{t}^* \right\|_p^p = |\alpha_0|^p |2 - t^*|^p$$

$$+ |\alpha_1|^p \cdot (|2 - t^*|^p + 2|t^*|^p)$$

$$+ |\alpha_2|^p \cdot (|-2 - t^*|^p + 2|t^*|^p)$$

$$+ |\alpha_3|^p \cdot |-2 - t^*|^p$$

# Isolating Parallelepipeds

Want to solve $M(t^*) \cdot \begin{pmatrix} |\alpha_0|^p \\ |\alpha_1|^p \\ \vdots \\ |\alpha_k|^p \end{pmatrix} = \begin{pmatrix} 1+\varepsilon \\ 1 \\ \vdots \\ 1 \end{pmatrix}$

# Isolating Parallelepipeds

Want to solve $M(t^*) \cdot \begin{pmatrix} |\alpha_0|^p \\ |\alpha_1|^p \\ \vdots \\ |\alpha_k|^p \end{pmatrix} = \begin{pmatrix} 1+\varepsilon \\ 1 \\ \vdots \\ 1 \end{pmatrix}$

- $M(t^*)$ is stochastic. I.e., if we set $\alpha_0 = \alpha_1 = \cdots = \alpha_k$, then the distances will all be the same.

# Isolating Parallelepipeds

Want to solve $M(t^*) \cdot \begin{pmatrix} |\alpha_0|^p \\ |\alpha_1|^p \\ \vdots \\ |\alpha_k|^p \end{pmatrix} = \begin{pmatrix} 1+\varepsilon \\ 1 \\ \vdots \\ 1 \end{pmatrix}$

- $M(t^*)$ is stochastic. I.e., if we set $\alpha_0 = \alpha_1 = \cdots = \alpha_k$, then the distances will all be the same.
- Therefore, it suffices to find $t^*$ such that $M(t^*)$ is non-singular.

# Isolating Parallelepipeds

Want to solve $M(t^*) \cdot \begin{pmatrix} |\alpha_0|^p \\ |\alpha_1|^p \\ \vdots \\ |\alpha_k|^p \end{pmatrix} = \begin{pmatrix} 1+\varepsilon \\ 1 \\ \vdots \\ 1 \end{pmatrix}$

- $M(t^*)$ is stochastic. I.e., if we set $\alpha_0 = \alpha_1 = \cdots = \alpha_k$, then the distances will all be the same.
- Therefore, it suffices to find $t^*$ such that $M(t^*)$ is non-singular.
- Entries in $M(t^*)$ look like $\sum_i a_i |b_i - t^*|^p$.

# Isolating Parallelepipeds

Want to solve $M(t^*) \cdot \begin{pmatrix} |\alpha_0|^p \\ |\alpha_1|^p \\ \vdots \\ |\alpha_k|^p \end{pmatrix} = \begin{pmatrix} 1+\varepsilon \\ 1 \\ \vdots \\ 1 \end{pmatrix}$

- $M(t^*)$ is stochastic. I.e., if we set $\alpha_0 = \alpha_1 = \cdots = \alpha_k$, then the distances will all be the same.
- Therefore, it suffices to find $t^*$ such that $M(t^*)$ is non-singular.
- Entries in $M(t^*)$ look like $\sum_i a_i |b_i - t^*|^p$.
  - Piecewise polynomial in $t^*$ when p is an integer.

# Isolating Parallelepipeds

Want to solve $M(t^*) \cdot \begin{pmatrix} |\alpha_0|^p \\ |\alpha_1|^p \\ \vdots \\ |\alpha_k|^p \end{pmatrix} = \begin{pmatrix} 1+\varepsilon \\ 1 \\ \vdots \\ 1 \end{pmatrix}$

- $M(t^*)$ is stochastic. I.e., if we set $\alpha_0 = \alpha_1 = \cdots = \alpha_k$, then the distances will all be the same.
- Therefore, it suffices to find $t^*$ such that $M(t^*)$ is non-singular.
- Entries in $M(t^*)$ look like $\sum_i a_i |b_i - t^*|^p$.
    - Piecewise polynomial in $t^*$ when p is an integer.
    - $\det(M(t^*))$ is a piecewise polynomial in $t^*$.

# Isolating Parallelepipeds

Want to solve $M(t^*) \cdot \begin{pmatrix} |\alpha_0|^p \\ |\alpha_1|^p \\ \vdots \\ |\alpha_k|^p \end{pmatrix} = \begin{pmatrix} 1+\varepsilon \\ 1 \\ \vdots \\ 1 \end{pmatrix}$

- $M(t^*)$ is stochastic. I.e., if we set $\alpha_0 = \alpha_1 = \cdots = \alpha_k$, then the distances will all be the same.
- Therefore, it suffices to find $t^*$ such that $M(t^*)$ is non-singular.
- Entries in $M(t^*)$ look like $\sum_i a_i |b_i - t^*|^p$.
  - Piecewise polynomial in $t^*$ when p is an integer.
  - $\det(M(t^*))$ is a piecewise polynomial in $t^*$ .
  - We show that it is not always the zero polynomial when p is odd.

# Also, Approximate CVP

Max-2-SAT reduction ⇒ hardness of $(1+\varepsilon)$-approx $\mathsf{CVP}_p$ for all $p$.

# Also, Approximate CVP

Max-2-SAT reduction $\Rightarrow$ hardness of $(1+\varepsilon)$-approx $\mathsf{CVP}_p$ for all $p$.

Formally GapETH hardness.

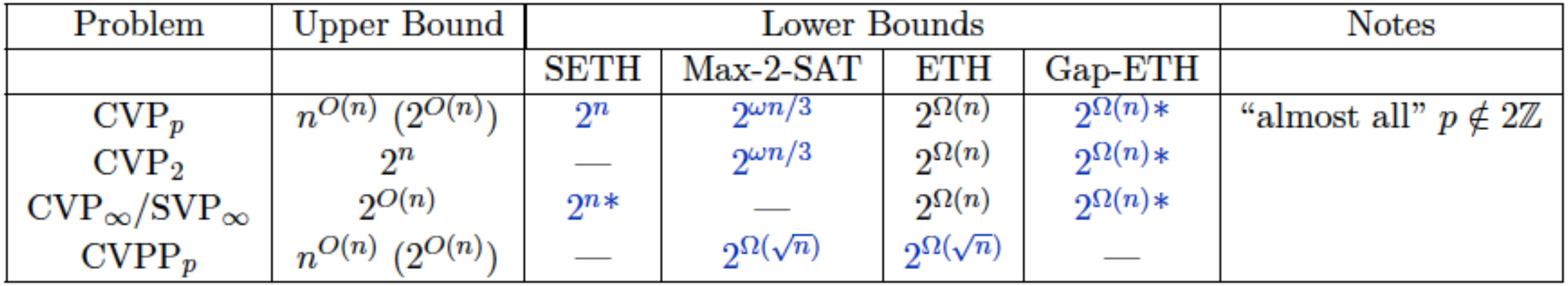
# Also, Approximate CVP

Max-2-SAT reduction $\Rightarrow$ hardness of $(1+\varepsilon)$-approx $\mathsf{CVP}_p$ for all $p$.

Formally GapETH hardness.

No $2^{o(n)}$-time for approx Max-2-SAT $\Rightarrow$ No $2^{o(n)}$-time for approx $\mathsf{CVP}_p$.

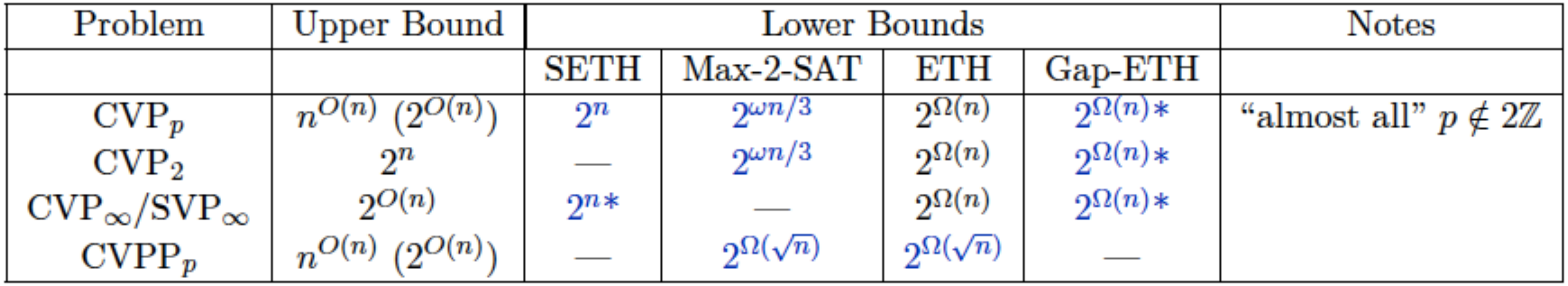
# Summary of Results



| Problem | Upper Bound | Lower Bounds | | | | Notes |
|---|---|---|---|---|---|---|
| | | SETH | Max-2-SAT | ETH | Gap-ETH | |
| $\mathrm{CVP}_p$ | $n^{O(n)}$ $(2^{O(n)})$ | $2^n$ | $2^{\omega n/3}$ | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | "almost all" $p \notin 2\mathbb{Z}$ |
| $\mathrm{CVP}_2$ | $2^n$ | — | $2^{\omega n/3}$ | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | |
| $\mathrm{CVP}_\infty/\mathrm{SVP}_\infty$ | $2^{O(n)}$ | $2^n*$ | — | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | |
| $\mathrm{CVPP}_p$ | $n^{O(n)}$ $(2^{O(n)})$ | — | $2^{\Omega(\sqrt{n})}$ | $2^{\Omega(\sqrt{n})}$ | — | |

Blue = new result.
(…) = approximation algorithm
* = hardness for some constant approximation factor

# Summary of Results

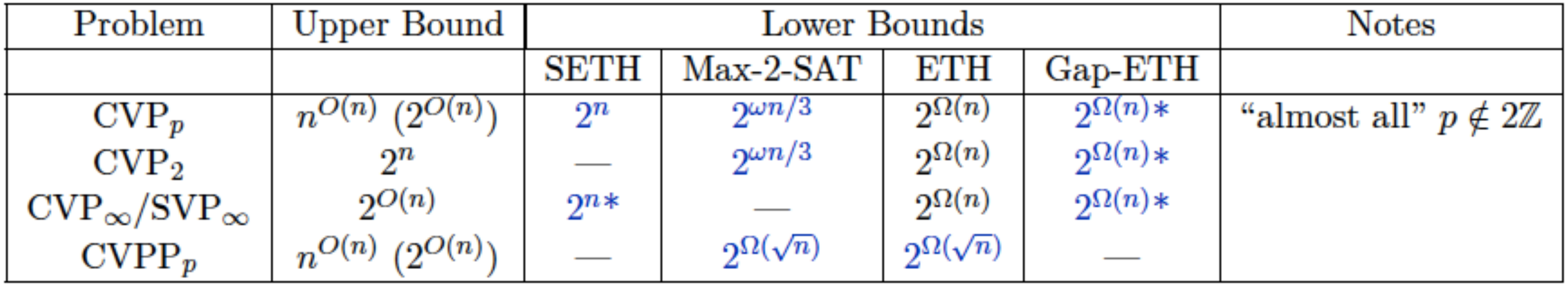


| Problem | Upper Bound | Lower Bounds | | | | Notes |
|---|---|---|---|---|---|---|
| | | SETH | Max-2-SAT | ETH | Gap-ETH | |
| $\mathrm{CVP}_p$ | $n^{O(n)}$ $(2^{O(n)})$ | $2^n$ | $2^{\omega n/3}$ | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | "almost all" $p \notin 2\mathbb{Z}$ |
| $\mathrm{CVP}_2$ | $2^n$ | — | $2^{\omega n/3}$ | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | |
| $\mathrm{CVP}_\infty/\mathrm{SVP}_\infty$ | $2^{O(n)}$ | $2^n*$ | — | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | |
| $\mathrm{CVPP}_p$ | $n^{O(n)}$ $(2^{O(n)})$ | — | $2^{\Omega(\sqrt{n})}$ | $2^{\Omega(\sqrt{n})}$ | — | |

Blue = new result.
(…) = approximation algorithm
* = hardness for some constant approximation factor

**Pros**

# Summary of Results



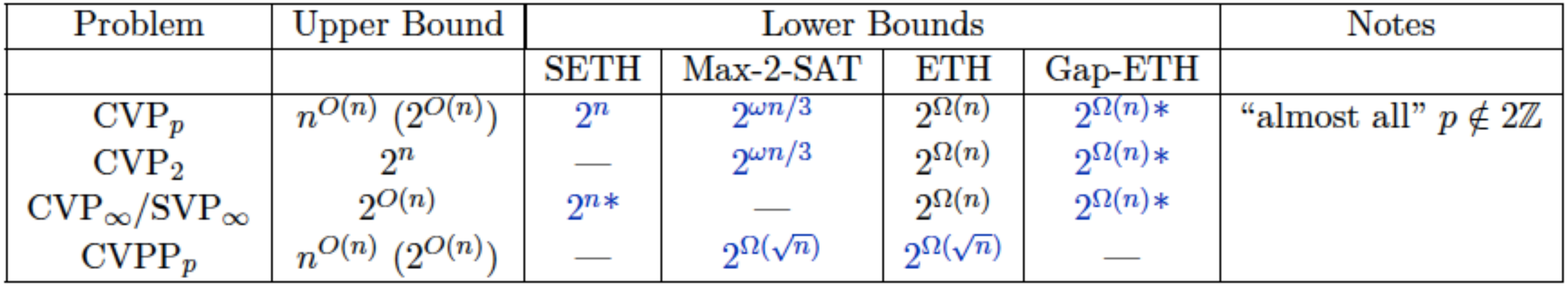
| Problem | Upper Bound | Lower Bounds | | | | Notes |
|---|---|---|---|---|---|---|
| | | SETH | Max-2-SAT | ETH | Gap-ETH | |
| $\mathrm{CVP}_p$ | $n^{O(n)}$ $(2^{O(n)})$ | $2^n$ | $2^{\omega n/3}$ | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | "almost all" $p \notin 2\mathbb{Z}$ |
| $\mathrm{CVP}_2$ | $2^n$ | — | $2^{\omega n/3}$ | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | |
| $\mathrm{CVP}_\infty/\mathrm{SVP}_\infty$ | $2^{O(n)}$ | $2^n*$ | — | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | |
| $\mathrm{CVPP}_p$ | $n^{O(n)}$ $(2^{O(n)})$ | — | $2^{\Omega(\sqrt{n})}$ | $2^{\Omega(\sqrt{n})}$ | — | |

Blue = new result.
(…) = approximation algorithm
* = hardness for some constant approximation factor

**<u>Pros</u>**

- We actually proved something!

# Summary of Results



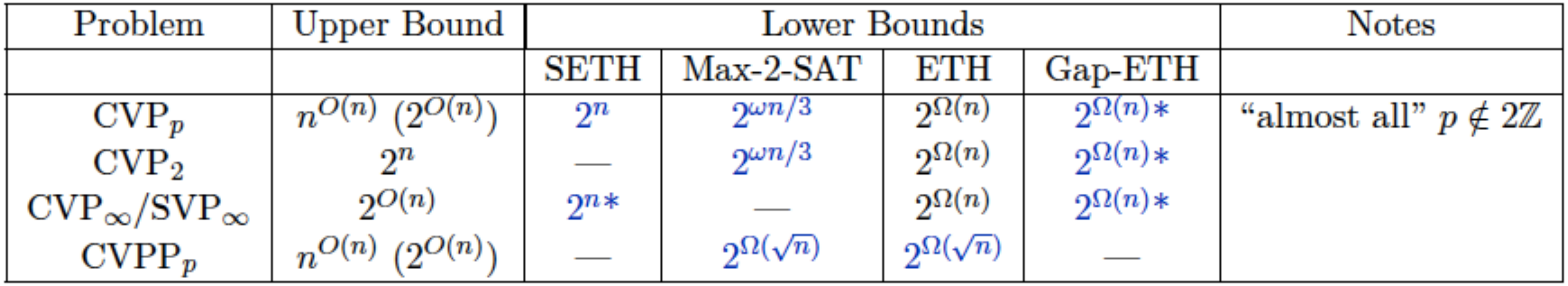
| Problem | Upper Bound | Lower Bounds | | | | Notes |
|---|---|---|---|---|---|---|
| | | SETH | Max-2-SAT | ETH | Gap-ETH | |
| $\mathrm{CVP}_p$ | $n^{O(n)}$ $(2^{O(n)})$ | $2^n$ | $2^{\omega n/3}$ | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | "almost all" $p \notin 2\mathbb{Z}$ |
| $\mathrm{CVP}_2$ | $2^n$ | — | $2^{\omega n/3}$ | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | |
| $\mathrm{CVP}_\infty/\mathrm{SVP}_\infty$ | $2^{O(n)}$ | $2^n*$ | — | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | |
| $\mathrm{CVPP}_p$ | $n^{O(n)}$ $(2^{O(n)})$ | — | $2^{\Omega(\sqrt{n})}$ | $2^{\Omega(\sqrt{n})}$ | — | |

Blue = new result.
(…) = approximation algorithm
* = hardness for some constant approximation factor

**Pros**

- We actually proved something!
- It's sort of tight!

# Summary of Results



| Problem | Upper Bound | Lower Bounds | | | | Notes |
|---|---|---|---|---|---|---|
| | | SETH | Max-2-SAT | ETH | Gap-ETH | |
| $\mathrm{CVP}_p$ | $n^{O(n)}$ $(2^{O(n)})$ | $2^n$ | $2^{\omega n/3}$ | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | "almost all" $p \notin 2\mathbb{Z}$ |
| $\mathrm{CVP}_2$ | $2^n$ | — | $2^{\omega n/3}$ | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | |
| $\mathrm{CVP}_\infty/\mathrm{SVP}_\infty$ | $2^{O(n)}$ | $2^n*$ | — | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | |
| $\mathrm{CVPP}_p$ | $n^{O(n)}$ $(2^{O(n)})$ | — | $2^{\Omega(\sqrt{n})}$ | $2^{\Omega(\sqrt{n})}$ | — | |

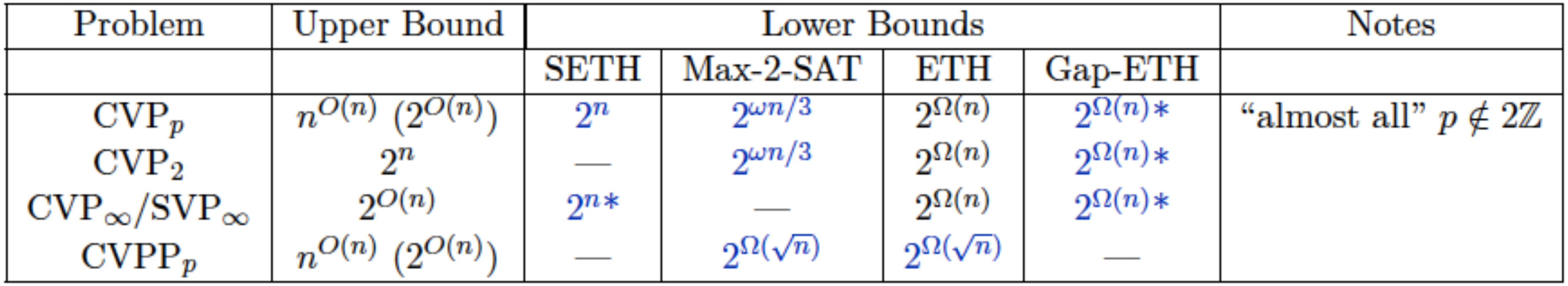
Blue = new result.
(…) = approximation algorithm
* = hardness for some constant approximation factor

**Pros**

- We actually proved something!
- It's sort of tight!

**Cons**

# Summary of Results



| Problem | Upper Bound | Lower Bounds | | | | Notes |
|---|---|---|---|---|---|---|
| | | SETH | Max-2-SAT | ETH | Gap-ETH | |
| $\mathrm{CVP}_p$ | $n^{O(n)}$ $(2^{O(n)})$ | $2^n$ | $2^{\omega n/3}$ | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | "almost all" $p \notin 2\mathbb{Z}$ |
| $\mathrm{CVP}_2$ | $2^n$ | — | $2^{\omega n/3}$ | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | |
| $\mathrm{CVP}_\infty/\mathrm{SVP}_\infty$ | $2^{O(n)}$ | $2^n*$ | — | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | |
| $\mathrm{CVPP}_p$ | $n^{O(n)}$ $(2^{O(n)})$ | — | $2^{\Omega(\sqrt{n})}$ | $2^{\Omega(\sqrt{n})}$ | — | |

Blue = new result.
(…) = approximation algorithm
* = hardness for some constant approximation factor

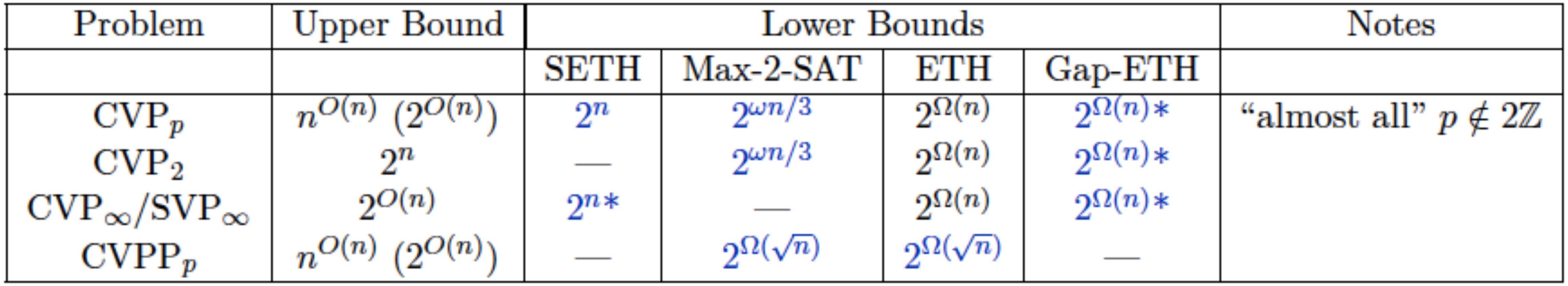
**Pros**

- We actually proved something!
- It's sort of tight!

**Cons**

- CVP, not SVP.

# Summary of Results



| Problem | Upper Bound | Lower Bounds | | | | Notes |
|---|---|---|---|---|---|---|
| | | SETH | Max-2-SAT | ETH | Gap-ETH | |
| $\mathrm{CVP}_p$ | $n^{O(n)}$ $(2^{O(n)})$ | $2^n$ | $2^{\omega n/3}$ | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | "almost all" $p \notin 2\mathbb{Z}$ |
| $\mathrm{CVP}_2$ | $2^n$ | — | $2^{\omega n/3}$ | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | |
| $\mathrm{CVP}_\infty/\mathrm{SVP}_\infty$ | $2^{O(n)}$ | $2^n*$ | — | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | |
| $\mathrm{CVPP}_p$ | $n^{O(n)}$ $(2^{O(n)})$ | — | $2^{\Omega(\sqrt{n})}$ | $2^{\Omega(\sqrt{n})}$ | — | |

Blue = new result.
(…) = approximation algorithm
* = hardness for some constant approximation factor

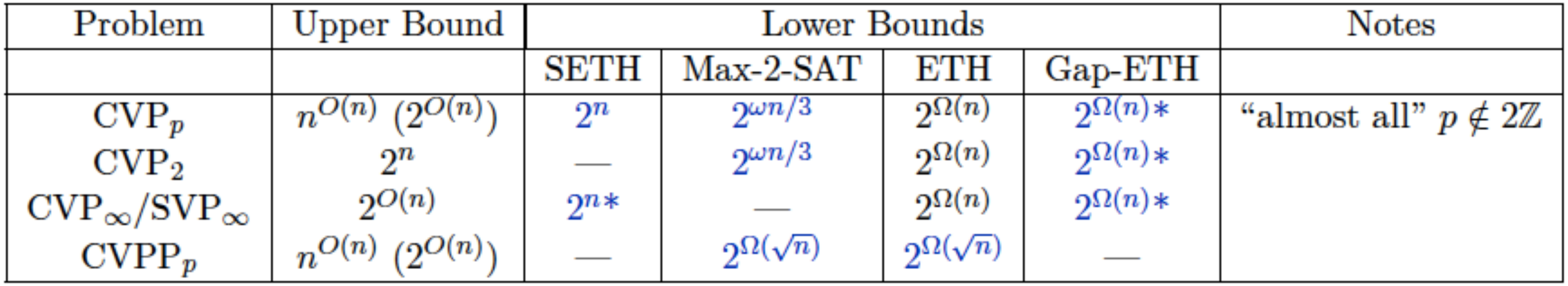
**Pros**

- We actually proved something!
- It's sort of tight!

**Cons**

- CVP, not SVP.
- Exact/near-exact CVP only.

# Summary of Results



| Problem | Upper Bound | Lower Bounds | | | | Notes |
|---|---|---|---|---|---|---|
| | | SETH | Max-2-SAT | ETH | Gap-ETH | |
| $\mathrm{CVP}_p$ | $n^{O(n)}$ $(2^{O(n)})$ | $2^n$ | $2^{\omega n/3}$ | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | "almost all" $p \notin 2\mathbb{Z}$ |
| $\mathrm{CVP}_2$ | $2^n$ | — | $2^{\omega n/3}$ | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | |
| $\mathrm{CVP}_\infty/\mathrm{SVP}_\infty$ | $2^{O(n)}$ | $2^n*$ | — | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | |
| $\mathrm{CVPP}_p$ | $n^{O(n)}$ $(2^{O(n)})$ | — | $2^{\Omega(\sqrt{n})}$ | $2^{\Omega(\sqrt{n})}$ | — | |

Blue = new result.
(…) = approximation algorithm
* = hardness for some constant approximation factor

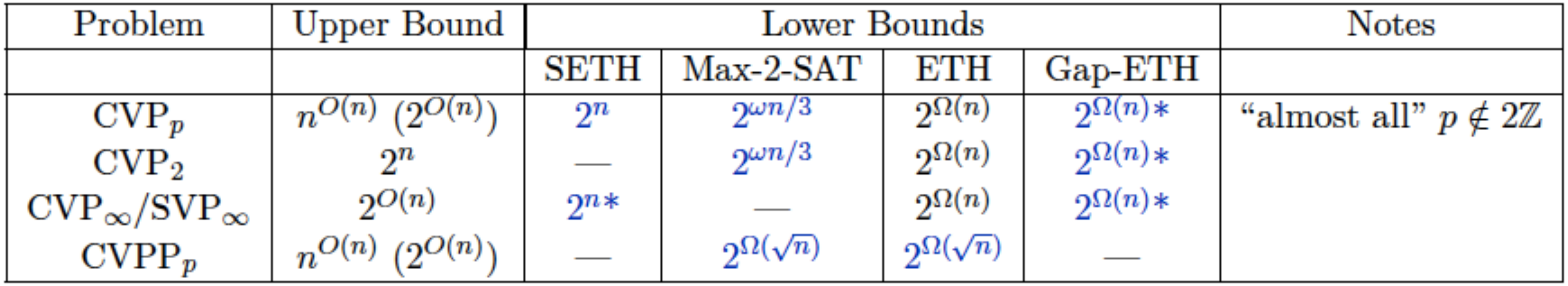
**Pros**

- We actually proved something!
- It's sort of tight!

**Cons**

- CVP, not SVP.
- Exact/near-exact CVP only.
- No $\ell_2$.

# Summary of Results



| Problem | Upper Bound | Lower Bounds | | | | Notes |
|---|---|---|---|---|---|---|
| | | SETH | Max-2-SAT | ETH | Gap-ETH | |
| $\mathrm{CVP}_p$ | $n^{O(n)}$ $(2^{O(n)})$ | $2^n$ | $2^{\omega n/3}$ | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | "almost all" $p \notin 2\mathbb{Z}$ |
| $\mathrm{CVP}_2$ | $2^n$ | — | $2^{\omega n/3}$ | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | |
| $\mathrm{CVP}_\infty/\mathrm{SVP}_\infty$ | $2^{O(n)}$ | $2^n*$ | — | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | |
| $\mathrm{CVPP}_p$ | $n^{O(n)}$ $(2^{O(n)})$ | — | $2^{\Omega(\sqrt{n})}$ | $2^{\Omega(\sqrt{n})}$ | — | |

Blue = new result.
(…) = approximation algorithm
* = hardness for some constant approximation factor

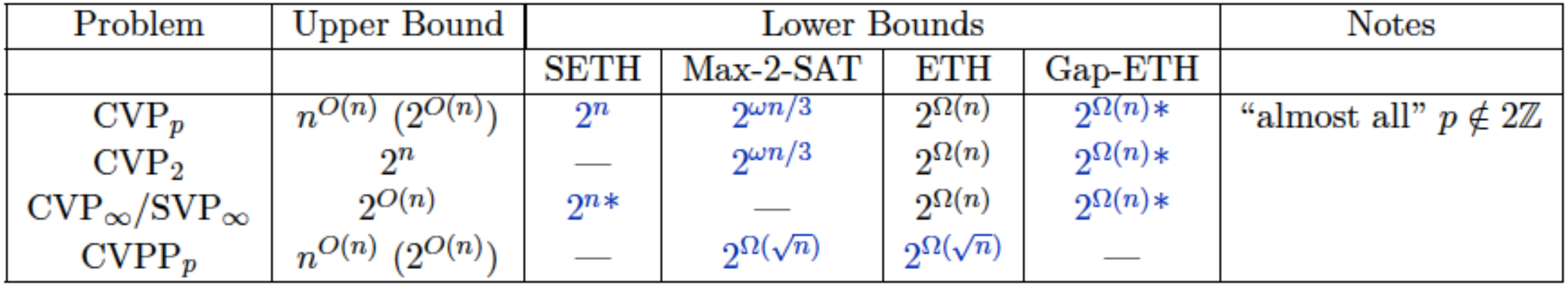
**Pros**

- We actually proved something!
- It's sort of tight!

**Cons**

- CVP, not SVP.
- Exact/near-exact CVP only.
- No $\ell_2$.
- Very artificial CVP instance.

# Summary of Results



| Problem | Upper Bound | Lower Bounds | | | | Notes |
|---|---|---|---|---|---|---|
| | | SETH | Max-2-SAT | ETH | Gap-ETH | |
| $\mathrm{CVP}_p$ | $n^{O(n)}$ $(2^{O(n)})$ | $2^n$ | $2^{\omega n/3}$ | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | "almost all" $p \notin 2\mathbb{Z}$ |
| $\mathrm{CVP}_2$ | $2^n$ | — | $2^{\omega n/3}$ | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | |
| $\mathrm{CVP}_\infty/\mathrm{SVP}_\infty$ | $2^{O(n)}$ | $2^n*$ | — | $2^{\Omega(n)}$ | $2^{\Omega(n)}*$ | |
| $\mathrm{CVPP}_p$ | $n^{O(n)}$ $(2^{O(n)})$ | — | $2^{\Omega(\sqrt{n})}$ | $2^{\Omega(\sqrt{n})}$ | — | |

Blue = new result.
(…) = approximation algorithm
* = hardness for some constant approximation factor

**Pros**

- We actually proved something!
- It's sort of tight!

**Cons**

- CVP, not SVP.
- Exact/near-exact CVP only.
- No $\ell_2$.
- Very artificial CVP instance.
- d >> n.

# Break?

# Act 3:
# What about SVP?

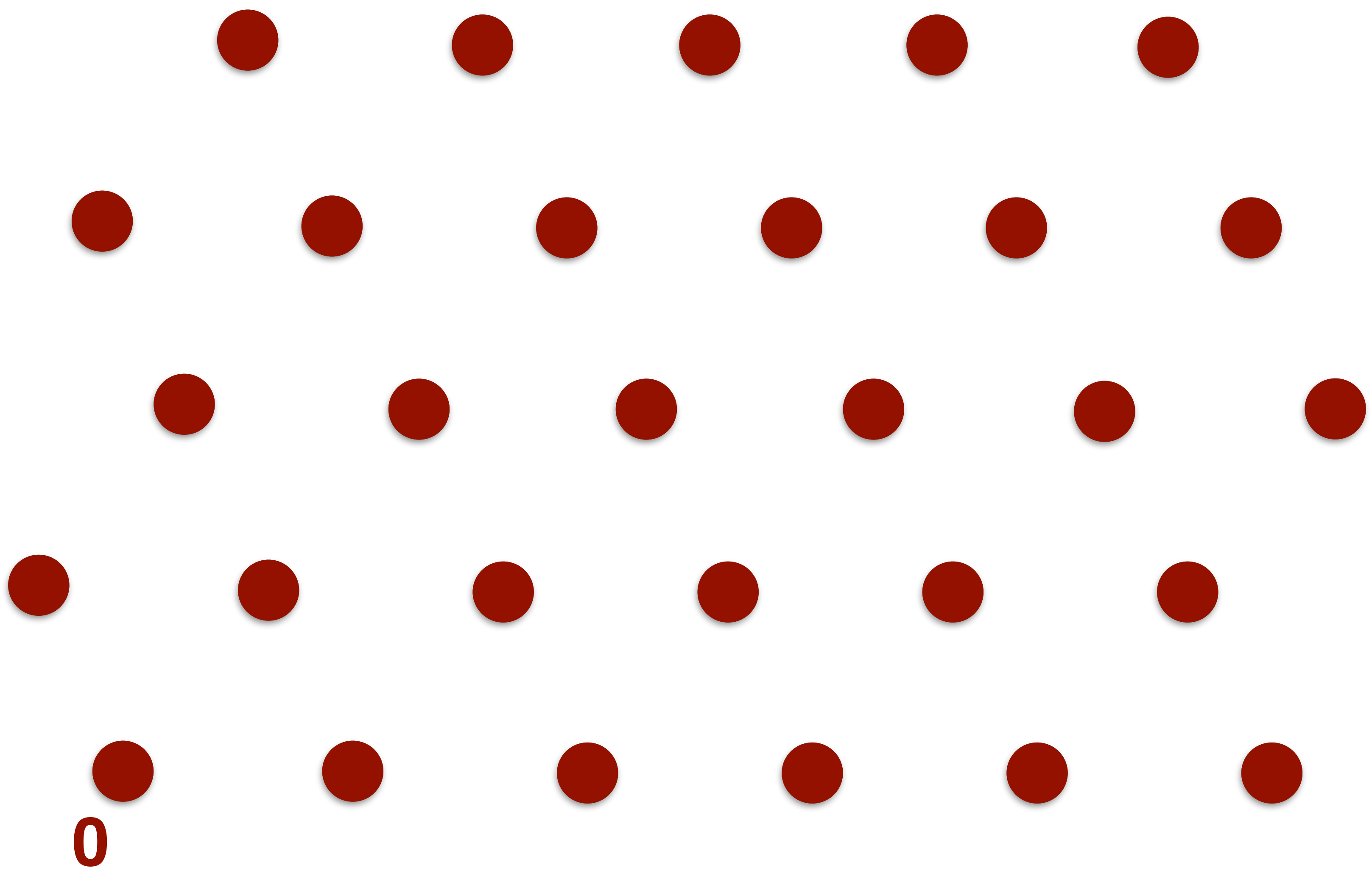
Divesh Aggarwal

Centre for Quantum Technologies

NUS National University of Singapore

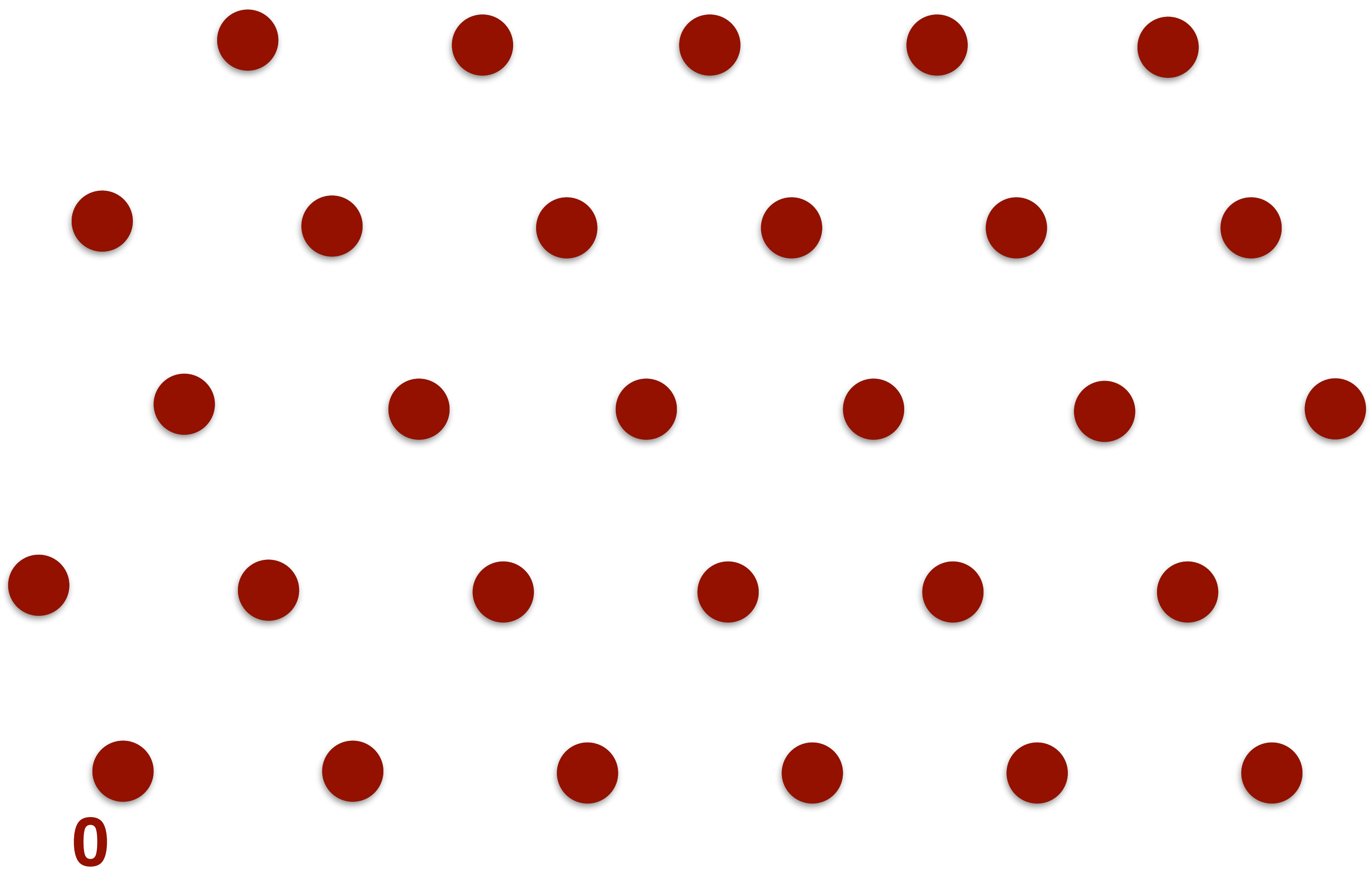
# The Shortest Vector Problem

# The Shortest Vector Problem



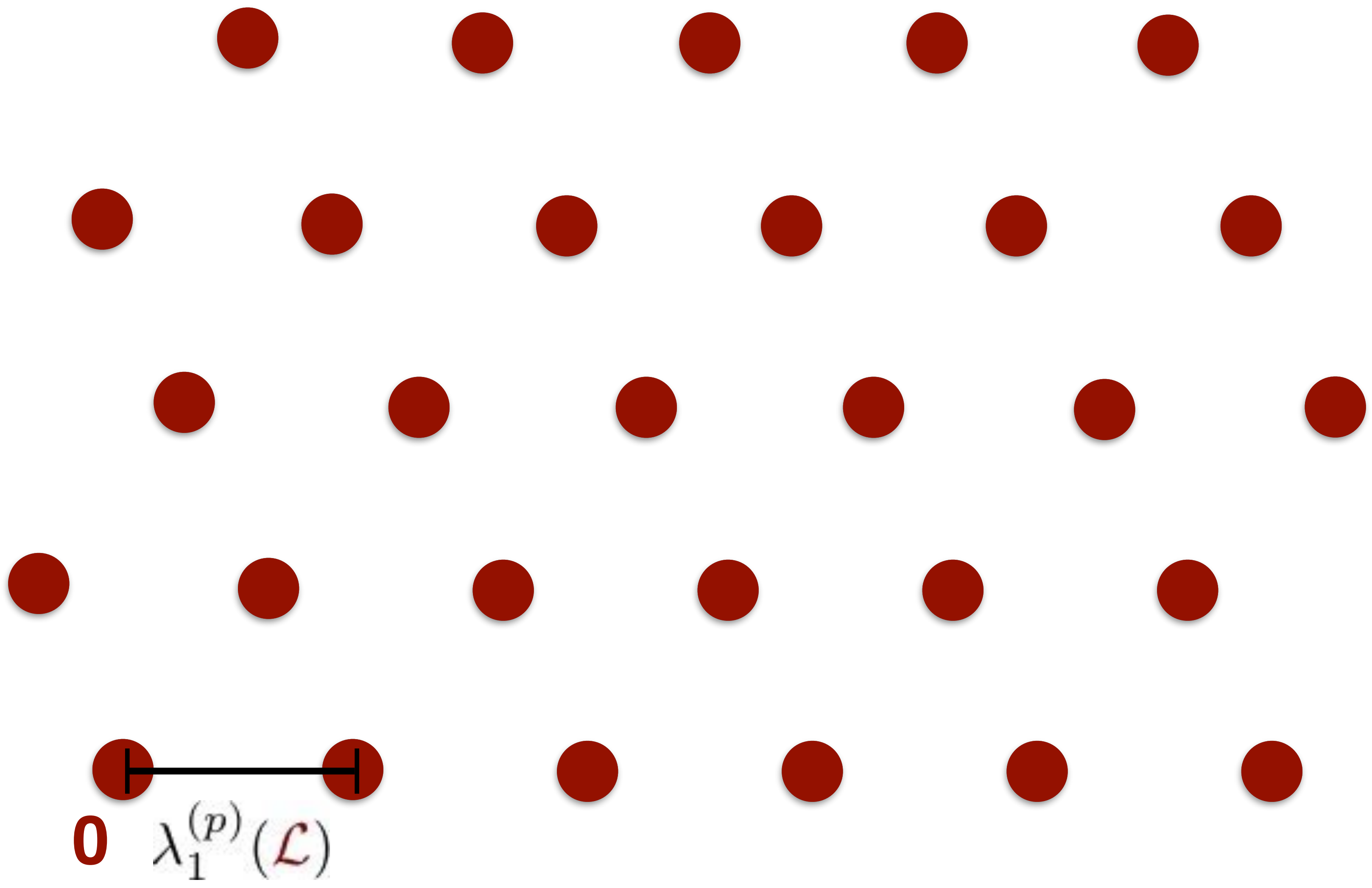
# The Shortest Vector Problem

$$\lambda_1^{(p)}(\mathcal{L}) := \min_{\mathbf{y} \in \mathcal{L}_{\neq \mathbf{0}}} \|\mathbf{y}\|_p$$



# The Shortest Vector Problem

$$\lambda_1^{(p)}(\mathcal{L}) := \min_{\mathbf{y} \in \mathcal{L}_{\neq \mathbf{0}}} \|\mathbf{y}\|_p$$



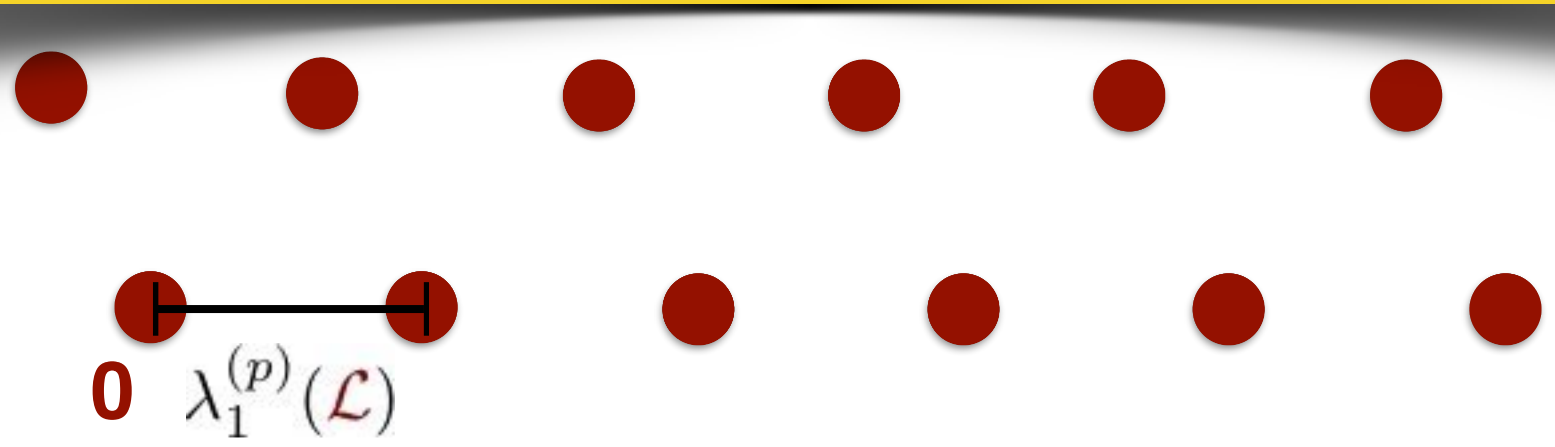
# The Shortest Vector Problem

$$\lambda_1^{(p)}(\mathcal{L}) := \min_{\mathbf{y} \in \mathcal{L}_{\neq \mathbf{0}}} \|\mathbf{y}\|_p$$

SVP$_p$ is the computational problem that asks us to compute $\lambda_1^{(p)}(\mathcal{L})$.

Approximate SVP$_p$ asks us to approximate $\lambda_1^{(p)}(\mathcal{L})$.

(We'll switch freely between the search and decision problems.)



# SVP Algorithms
# (it's complicated…)

| p | | |
|---|---|---|
| All | $2^{O(n)}$ | [AKS01, BN09, AJ08, DPV11] |
| 2 | $2^{n+o(n)}$ | [ADRS15, AS18] |
| 2 | $2^{n/2+o(n)}$ | 2-approx [ADRS15] |
| 2 | $n^{O(n)}$ (but fast) | (n=150!) [KT17] |
| 2 | $(3/2)^{n/2+o(n)} \approx 2^{0.29n}$ | Heuristic [BDGL15] |
| ∞ | $\approx 3^d$ | [DM18] |
| ∞ | $2^{0.62d}$ | Heuristic [DM18] |

# Hardness of SVP (it's hard…)

# Hardness of SVP (it's hard…)

- Van Emde Boaz asked in 1981 whether SVP is NP-hard.

# Hardness of SVP (it's hard…)

- Van Emde Boaz asked in 1981 whether SVP is NP-hard.
- Answered in 1998 by Ajtai. (Yes.)

# Hardness of SVP
# (it's hard…)

- Van Emde Boaz asked in 1981 whether SVP is NP-hard.
- Answered in 1998 by Ajtai. (Yes.)
- NP-hard to approximate to within any constant [CN98, Mic01, Kho05]

# Hardness of SVP (it’s hard…)

- Van Emde Boaz asked in 1981 whether SVP is NP-hard.
- Answered in 1998 by Ajtai. (Yes.)
- NP-hard to approximate to within any constant [CN98, Mic01, Kho05]
- “Hard” to approximate to “near-polynomial” factors […, HR12]

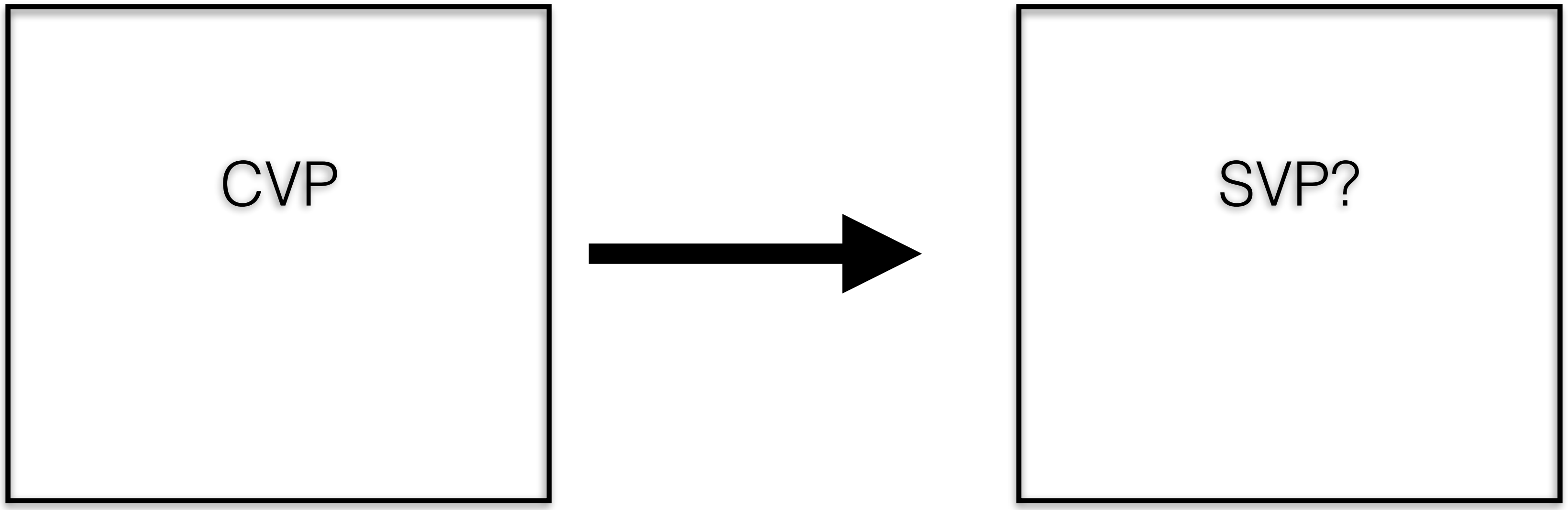
# Hardness of SVP
# (it's hard…)

- Van Emde Boaz asked in 1981 whether SVP is NP-hard.
- Answered in 1998 by Ajtai. (Yes.)
- NP-hard to approximate to within any constant [CN98, Mic01, Kho05]
- "Hard" to approximate to "near-polynomial" factors […, HR12]
  - $\gamma = n^{c/\log\log n}$

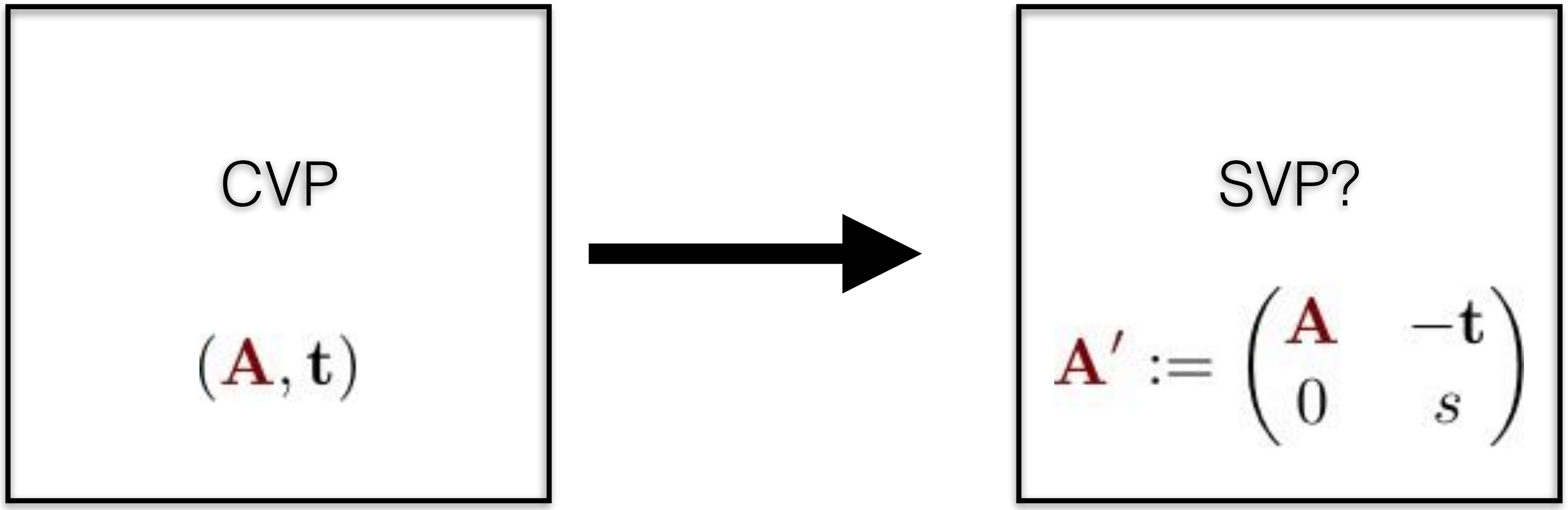
# Hardness of SVP
# (it's hard…)

- Van Emde Boaz asked in 1981 whether SVP is NP-hard.
- Answered in 1998 by Ajtai. (Yes.)
- NP-hard to approximate to within any constant [CN98, Mic01, Kho05]
- "Hard" to approximate to "near-polynomial" factors […, HR12]
    - $\gamma = n^{c/\log\log n}$
- All known reductions are randomized [Mic12]

# Hardness of SVP: Dream Proof



# Hardness of SVP: Dream Proof



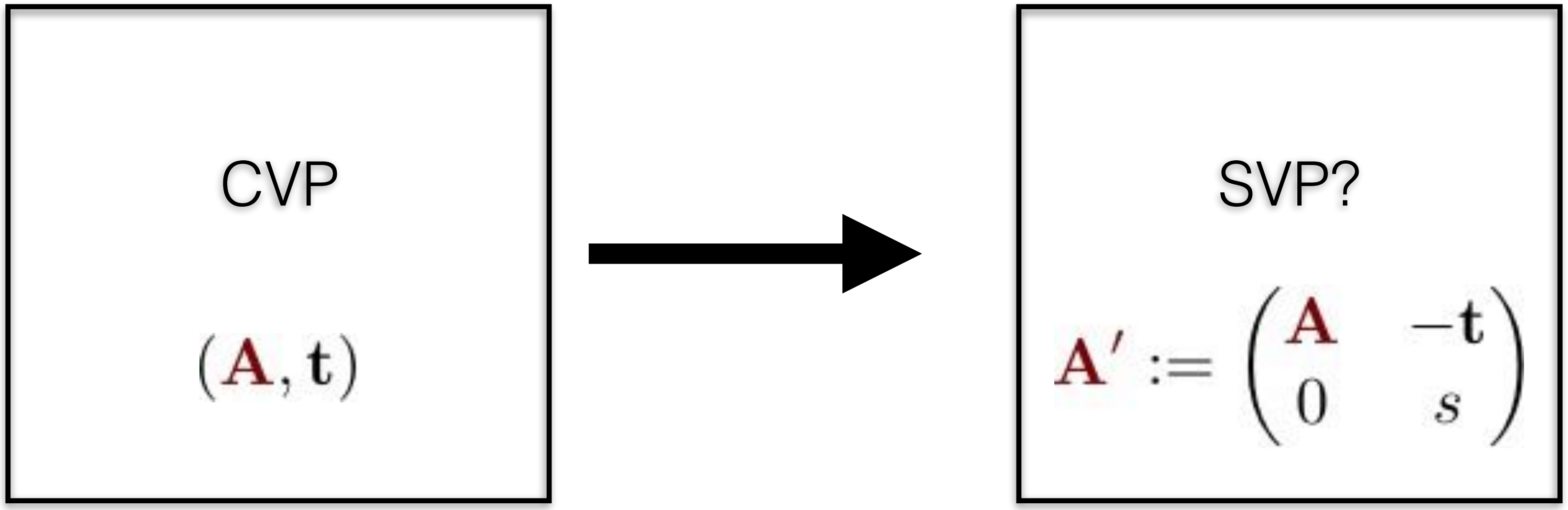
CVP

$(\mathbf{A}, \mathbf{t})$

SVP?

$$\mathbf{A}' := \begin{pmatrix} \mathbf{A} & -\mathbf{t} \\ 0 & s \end{pmatrix}$$

# Hardness of SVP: Dream Proof



CVP

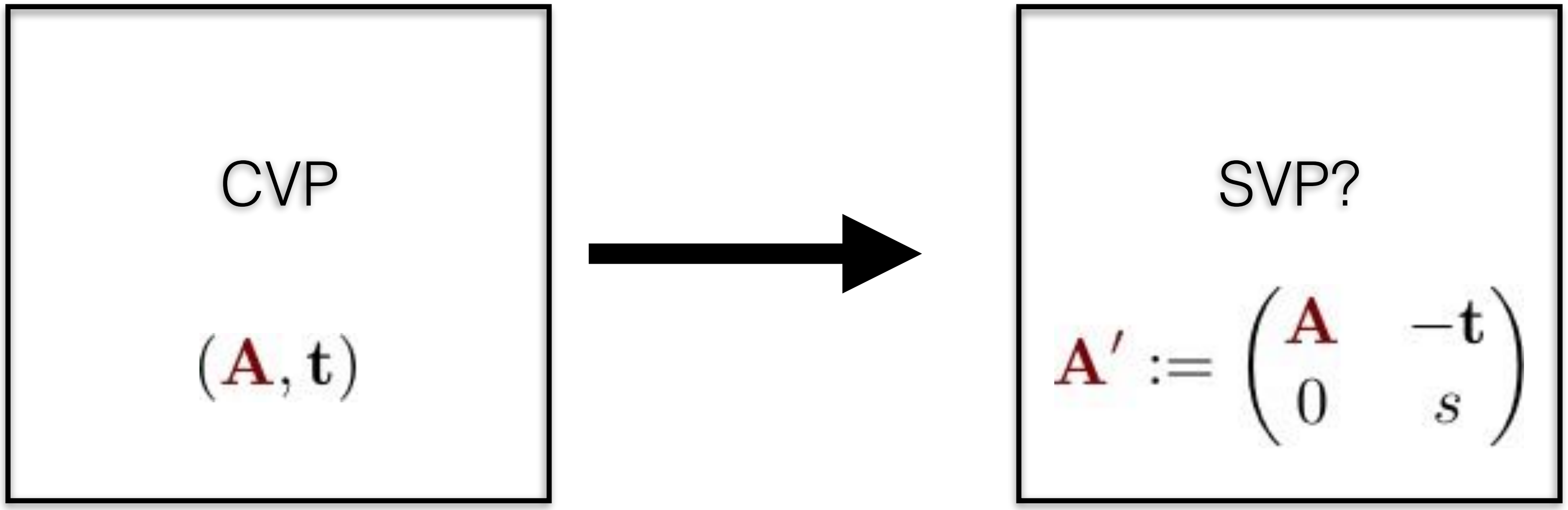
$(\mathbf{A}, \mathbf{t})$

SVP?

$$\mathbf{A}' := \begin{pmatrix} \mathbf{A} & -\mathbf{t} \\ 0 & s \end{pmatrix}$$

$$\mathcal{L}(\mathbf{A}') = \{(\mathbf{y} - k\mathbf{t}, ks) \ : \ \mathbf{y} \in \mathcal{L}(\mathbf{A}), \ k \in \mathbb{Z}\}$$

# Hardness of SVP: Dream Proof



CVP

$(\mathbf{A}, \mathbf{t})$

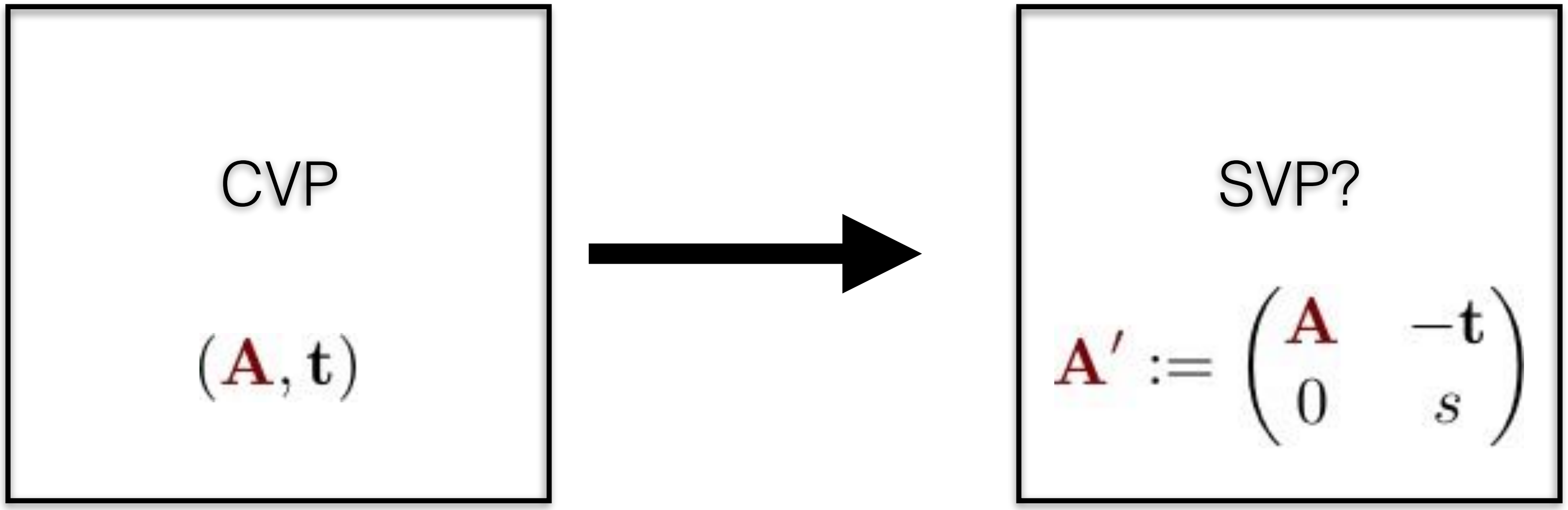
SVP?

$$\mathbf{A}' := \begin{pmatrix} \mathbf{A} & -\mathbf{t} \\ 0 & s \end{pmatrix}$$

$$\mathcal{L}(\mathbf{A}') = \{(\mathbf{y} - k\mathbf{t}, ks) \ : \ \mathbf{y} \in \mathcal{L}(\mathbf{A}),\ k \in \mathbb{Z}\}$$

For $\mathbf{y} \in \mathcal{L}(\mathbf{A})$, $(\mathbf{y} - \mathbf{t}, s) \in \mathcal{L}(\mathbf{A}')$ with $\|(\mathbf{y} - \mathbf{t}, s)\|_p^p = s^p + \|\mathbf{y} - \mathbf{t}\|^p$.

# Hardness of SVP: Dream Proof



CVP

$(\mathbf{A}, \mathbf{t})$

SVP?

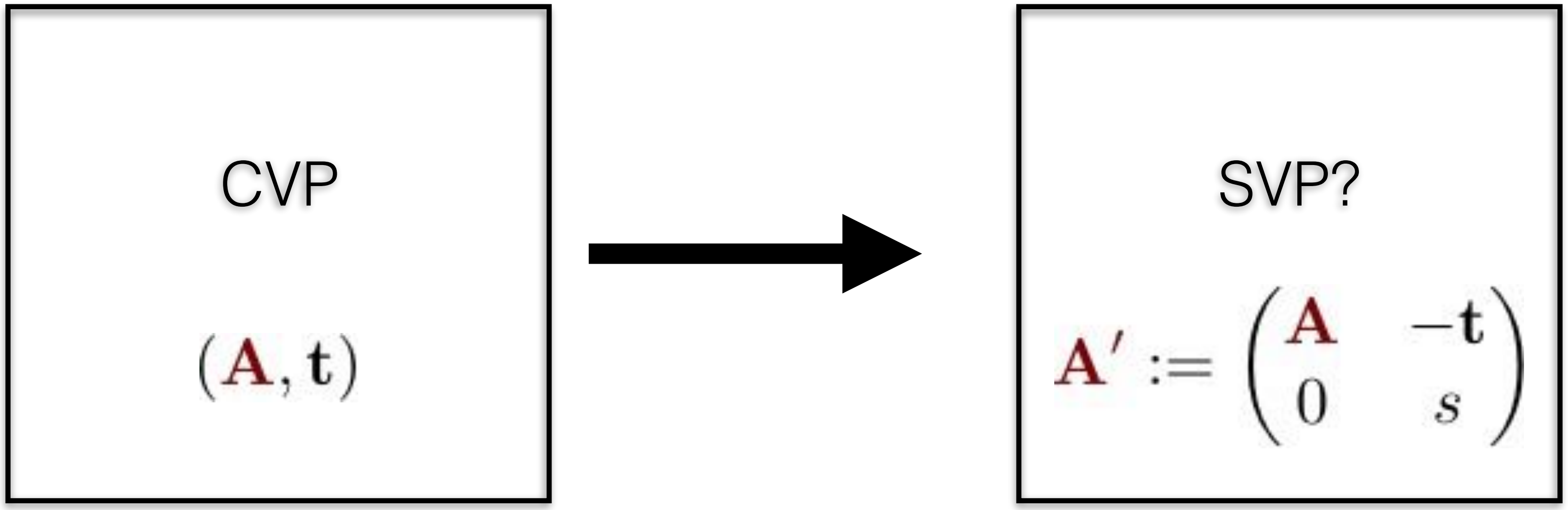
$$\mathbf{A}' := \begin{pmatrix} \mathbf{A} & -\mathbf{t} \\ 0 & s \end{pmatrix}$$

$$\mathcal{L}(\mathbf{A}') = \{(\mathbf{y} - k\mathbf{t}, ks) \ : \ \mathbf{y} \in \mathcal{L}(\mathbf{A}), \ k \in \mathbb{Z}\}$$

For $\mathbf{y} \in \mathcal{L}(\mathbf{A})$, $(\mathbf{y} - \mathbf{t}, s) \in \mathcal{L}(\mathbf{A}')$ with $\|(\mathbf{y} - \mathbf{t}, s)\|_p^p = s^p + \|\mathbf{y} - \mathbf{t}\|^p$.

Problem: Short vectors $(\mathbf{y} - k\mathbf{t}, ks) \in \mathcal{L}(\mathbf{A}')$ for $k \neq \pm 1$.

# Hardness of SVP: Dream Proof



CVP

$(\mathbf{A}, \mathbf{t})$

SVP?

$$\mathbf{A}' := \begin{pmatrix} \mathbf{A} & -\mathbf{t} \\ 0 & s \end{pmatrix}$$

$$\mathcal{L}(\mathbf{A}') = \{(\mathbf{y} - k\mathbf{t}, ks) \ : \ \mathbf{y} \in \mathcal{L}(\mathbf{A}), \ k \in \mathbb{Z}\}$$

For $\mathbf{y} \in \mathcal{L}(\mathbf{A})$, $(\mathbf{y} - \mathbf{t}, s) \in \mathcal{L}(\mathbf{A}')$ with $\|(\mathbf{y} - \mathbf{t}, s)\|_p^p = s^p + \|\mathbf{y} - \mathbf{t}\|^p$.

Problem: Short vectors $(\mathbf{y} - k\mathbf{t}, ks) \in \mathcal{L}(\mathbf{A}')$ for $k \neq \pm 1$.

Really only $k = 0$ is a problem. I.e., short vectors in $\mathcal{L}(\mathbf{A})$.

# Hardness of SVP: Dream Proof

$$\mathbf{A}' := \begin{pmatrix} \mathbf{A} & -\mathbf{t} \\ 0 & s \end{pmatrix}$$

Problem: Maybe $\lambda_1^{(p)}(\mathcal{L}(\mathbf{A})) < \mathrm{dist}_p(\mathbf{t}, \mathcal{L}(\mathbf{A}))$.

# Hardness of SVP: Dream Proof

$$\mathbf{A}' := \begin{pmatrix} \mathbf{A} & -\mathbf{t} \\ 0 & s \end{pmatrix}$$

Problem: Maybe $\lambda_1^{(p)}(\mathcal{L}(\mathbf{A})) < \mathrm{dist}_p(\mathbf{t}, \mathcal{L}(\mathbf{A}))$.

Ideal solution: Just assume that this doesn't happen.

# Hardness of SVP: Dream Proof

$$\mathbf{A}' := \begin{pmatrix} \mathbf{A} & -\mathbf{t} \\ 0 & s \end{pmatrix}$$

Problem: Maybe $\lambda_1^{(p)}(\mathcal{L}(\mathbf{A})) < \mathrm{dist}_p(\mathbf{t}, \mathcal{L}(\mathbf{A}))$.

Ideal solution: Just assume that this doesn't happen.

Real solution: Assume that it doesn't happen "often."

# Hardness of SVP: Dream Proof

$$\mathbf{A}' := \begin{pmatrix} \mathbf{A} & -\mathbf{t} \\ 0 & s \end{pmatrix}$$

Problem: Maybe $\lambda_1^{(p)}(\mathcal{L}(\mathbf{A})) < \mathrm{dist}_p(\mathbf{t}, \mathcal{L}(\mathbf{A}))$.

Ideal solution: Just assume that this doesn't happen.

Real solution: Assume that it doesn't happen "often."

$N_p(\mathcal{L}, r) := |\{\mathbf{y} \in \mathcal{L} \ : \ \|\mathbf{y}\|_p \leq r\}|$ (Number of short vectors.)

# Hardness of SVP: Dream Proof

$$\mathbf{A}' := \begin{pmatrix} \mathbf{A} & -\mathbf{t} \\ 0 & s \end{pmatrix}$$

Problem: Maybe $\lambda_1^{(p)}(\mathcal{L}(\mathbf{A})) < \mathrm{dist}_p(\mathbf{t}, \mathcal{L}(\mathbf{A}))$.

Ideal solution: Just assume that this doesn't happen.

Real solution: Assume that it doesn't happen "often."

$N_p(\mathcal{L}, r) := |\{\mathbf{y} \in \mathcal{L} \ : \ \|\mathbf{y}\|_p \leq r\}|$ (Number of short vectors.)

$N_p(\mathcal{L}, r; \mathbf{t}) := |\{\mathbf{y} \in \mathcal{L} \ : \ \|\mathbf{y} - \mathbf{t}\|_p \leq r\}|$ (Number of close vectors.)

# Hardness of SVP: Dream Proof

$$\mathbf{A}' := \begin{pmatrix} \mathbf{A} & -\mathbf{t} \\ 0 & s \end{pmatrix}$$

Problem: Maybe $\lambda_1^{(p)}(\mathcal{L}(\mathbf{A})) < \mathrm{dist}_p(\mathbf{t}, \mathcal{L}(\mathbf{A}))$.

Ideal solution: Just assume that this doesn't happen.

Real solution: Assume that it doesn't happen "often."

$N_p(\mathcal{L}, r) := |\{\mathbf{y} \in \mathcal{L} \ : \ \|\mathbf{y}\|_p \leq r\}|$ (Number of short vectors.)

$N_p(\mathcal{L}, r; \mathbf{t}) := |\{\mathbf{y} \in \mathcal{L} \ : \ \|\mathbf{y} - \mathbf{t}\|_p \leq r\}|$ (Number of close vectors.)

$$N_p(\mathcal{L}(\mathbf{A}), r) \ll N_p(\mathcal{L}(\mathbf{A}), r; \mathbf{t})$$

# Hardness of SVP: Dream Proof

$$\mathbf{A}' := \begin{pmatrix} \mathbf{A} & -\mathbf{t} \\ 0 & s \end{pmatrix}$$

Problem: Maybe $\lambda_1^{(p)}(\mathcal{L}(\mathbf{A})) < \mathrm{dist}_p(\mathbf{t}, \mathcal{L}(\mathbf{A}))$.

Ideal solution: Just assume that this doesn't happen.

Real solution: Assume that it doesn't happen "often."

$$N_p(\mathcal{L}, r) := |\{\mathbf{y} \in \mathcal{L} \ : \ \|\mathbf{y}\|_p \le r\}| \qquad \text{(Number of short vectors.)}$$

$$N_p(\mathcal{L}, r; \mathbf{t}) := |\{\mathbf{y} \in \mathcal{L} \ : \ \|\mathbf{y} - \mathbf{t}\|_p \le r\}| \qquad \text{(Number of close vectors.)}$$

$$N_p(\mathcal{L}(\mathbf{A}), r) \ll N_p(\mathcal{L}(\mathbf{A}), r; \mathbf{t})$$

"Many more close vectors than short vectors."

# Sparsification [Khot05]

$$\mathbf{A}' := \begin{pmatrix} \mathbf{A} & -\mathbf{t} \\ 0 & s \end{pmatrix}$$

“Many more close vectors than short vectors.”

# Sparsification [Khot05]

$$\mathbf{A}' := \begin{pmatrix} \mathbf{A} & -\mathbf{t} \\ 0 & s \end{pmatrix}$$

"Many more close vectors than short vectors."

$$\mathcal{L}' := \{\mathbf{A}\mathbf{z} \ : \ \mathbf{z} \in \mathbb{Z}^{n+1},\ x_1 z_1 + \cdots + x_{n+1} z_{n+1} \equiv 0 \bmod q\}$$

# Sparsification [Khot05]

$$\mathbf{A}' := \begin{pmatrix} \mathbf{A} & -\mathbf{t} \\ 0 & s \end{pmatrix}$$

"Many more close vectors than short vectors."

$$\mathcal{L}' := \{\mathbf{A}\mathbf{z} \ : \ \mathbf{z} \in \mathbb{Z}^{n+1},\ x_1 z_1 + \cdots + x_{n+1} z_{n+1} \equiv 0 \bmod q\}$$

$$\mathbf{x} \sim \mathbb{Z}_q^{n+1}$$

# Sparsification [Khot05]

$$\mathbf{A}' := \begin{pmatrix} \mathbf{A} & -\mathbf{t} \\ 0 & s \end{pmatrix}$$

“Many more close vectors than short vectors.”

$$\mathcal{L}' := \{\mathbf{Az} \ : \ \mathbf{z} \in \mathbb{Z}^{n+1},\ x_1 z_1 + \cdots + x_{n+1} z_{n+1} \equiv 0 \bmod q\}$$

$$\mathbf{x} \sim \mathbb{Z}_q^{n+1}$$

$$q \approx N_p(\mathcal{L}(\mathbf{A}), r; \mathbf{t})$$

# Sparsification [Khot05]

$$\mathbf{A}' := \begin{pmatrix} \mathbf{A} & -\mathbf{t} \\ 0 & s \end{pmatrix}$$

"Many more close vectors than short vectors."

$$\mathcal{L}' := \{\mathbf{Az} \ : \ \mathbf{z} \in \mathbb{Z}^{n+1},\ x_1 z_1 + \cdots + x_{n+1} z_{n+1} \equiv 0 \bmod q\}$$

$$\mathbf{x} \sim \mathbb{Z}_q^{n+1}$$

$$q \approx N_p(\mathcal{L}(\mathbf{A}), r; \mathbf{t})$$

Vectors in $\mathcal{L}'$ "look" independent with $\Pr[\mathbf{y} \in \mathcal{L}' \mid \mathbf{y} \in \mathcal{L}(\mathbf{A}')] = 1/q$.

# Sparsification [Khot05]

$$\mathbf{A}' := \begin{pmatrix} \mathbf{A} & -\mathbf{t} \\ 0 & s \end{pmatrix}$$

"Many more close vectors than short vectors."

$$\mathcal{L}' := \{\mathbf{Az} \ : \ \mathbf{z} \in \mathbb{Z}^{n+1},\ x_1 z_1 + \cdots + x_{n+1} z_{n+1} \equiv 0 \bmod q\}$$

If the initial CVP instance has many more close vectors than short vectors, then the shortest vector in $\mathcal{L}'$ will "correspond to" a close vector with high probability.

$$q \approx N_p(\mathcal{L}(\mathbf{A}), r; \mathbf{t})$$

Vectors in $\mathcal{L}'$ "look" independent with $\Pr[\mathbf{y} \in \mathcal{L}' \mid \mathbf{y} \in \mathcal{L}(\mathbf{A}')] = 1/q$.

# Sparsification [Khot05]

$$\mathbf{A}' := \begin{pmatrix} \mathbf{A} & -\mathbf{t} \\ 0 & s \end{pmatrix}$$

"Many more close vectors than short vectors."

$$\mathcal{L}' := \{\mathbf{A}\mathbf{z} \ : \ \mathbf{z} \in \mathbb{Z}^{n+1}, \ x_1 z_1 + \cdots + x_{n+1} z_{n+1} \equiv 0 \bmod q\}$$

If the initial CVP instance has many more close vectors than short vectors, then the shortest vector in $\mathcal{L}'$ will "correspond to" a close vector with high probability.

$$q \approx N_p(\mathcal{L}(\mathbf{A}), r; \mathbf{t})$$

It suffices to show hardness of CVP with more close vectors than short vectors.

(Note: The resulting lattice looks a lot like the lattices used in cryptography.)

# Increasing the Ratio of Close Vectors to Short Vectors

# Increasing the Ratio of Close Vectors to Short Vectors

Gadget: $(\mathbf{A}^\dagger, \mathbf{t}^\dagger)$

# Increasing the Ratio of Close Vectors to Short Vectors

Gadget: $(\mathbf{A}^\dagger, \mathbf{t}^\dagger)$

$$\widehat{\mathcal{L}} := \mathcal{L}\begin{pmatrix} \mathbf{A} & 0 \\ 0 & \mathbf{A}^\dagger \end{pmatrix} \qquad \widehat{\mathbf{t}} := \begin{pmatrix} \mathbf{t} \\ \mathbf{t}^\dagger \end{pmatrix}$$

# Increasing the Ratio of Close Vectors to Short Vectors

Gadget: $(\mathbf{A}^\dagger, \mathbf{t}^\dagger)$

$$\widehat{\mathcal{L}} := \mathcal{L}\begin{pmatrix} \mathbf{A} & 0 \\ 0 & \mathbf{A}^\dagger \end{pmatrix} \qquad \widehat{\mathbf{t}} := \begin{pmatrix} \mathbf{t} \\ \mathbf{t}^\dagger \end{pmatrix}$$

$$\widehat{\mathcal{L}} = \{(\mathbf{y}, \mathbf{y}^\dagger) \ : \ \mathbf{y} \in \mathcal{L}(\mathbf{A}),\ \mathbf{y}^\dagger \in \mathcal{L}(\mathbf{A}^\dagger)\}$$

# Increasing the Ratio of Close Vectors to Short Vectors

Gadget: $(\mathbf{A}^\dagger, \mathbf{t}^\dagger)$

$$\widehat{\mathcal{L}} := \mathcal{L}\begin{pmatrix} \mathbf{A} & 0 \\ 0 & \mathbf{A}^\dagger \end{pmatrix} \qquad \widehat{\mathbf{t}} := \begin{pmatrix} \mathbf{t} \\ \mathbf{t}^\dagger \end{pmatrix}$$

$$\widehat{\mathcal{L}} = \{(\mathbf{y}, \mathbf{y}^\dagger) \ : \ \mathbf{y} \in \mathcal{L}(\mathbf{A}), \ \mathbf{y}^\dagger \in \mathcal{L}(\mathbf{A}^\dagger)\}$$

$$N_p(\widehat{\mathcal{L}}, (r^p + (r^\dagger)^p)^{1/p}) \approx N_p(\mathcal{L}(\mathbf{A}), r) \cdot N_p(\mathcal{L}(\mathbf{A}^\dagger), r^\dagger)$$

# Increasing the Ratio of Close Vectors to Short Vectors

Gadget: $(\mathbf{A}^\dagger, \mathbf{t}^\dagger)$

$$\widehat{\mathcal{L}} := \mathcal{L}\begin{pmatrix} \mathbf{A} & 0 \\ 0 & \mathbf{A}^\dagger \end{pmatrix} \qquad \widehat{\mathbf{t}} := \begin{pmatrix} \mathbf{t} \\ \mathbf{t}^\dagger \end{pmatrix}$$

$$\widehat{\mathcal{L}} = \{(\mathbf{y}, \mathbf{y}^\dagger) \;:\; \mathbf{y} \in \mathcal{L}(\mathbf{A}),\ \mathbf{y}^\dagger \in \mathcal{L}(\mathbf{A}^\dagger)\}$$

$$N_p(\widehat{\mathcal{L}}, (r^p + (r^\dagger)^p)^{1/p}) \approx N_p(\mathcal{L}(\mathbf{A}), r) \cdot N_p(\mathcal{L}(\mathbf{A}^\dagger), r^\dagger)$$

$$N_p(\widehat{\mathcal{L}}, (r^p + (r^\dagger)^p)^{1/p}; \widehat{\mathbf{t}}) \approx N_p(\mathcal{L}(\mathbf{A}), r; \mathbf{t}) \cdot N_p(\mathcal{L}(\mathbf{A}^\dagger), r^\dagger; \mathbf{t}^\dagger)$$

# Increasing the Ratio of Close Vectors to Short Vectors

In order to get "many more close vectors than short vectors", we want

$$\frac{N_p(\mathcal{L}(\mathbf{A}^\dagger), r^\dagger; \mathbf{t}^\dagger)}{N_p(\mathcal{L}(\mathbf{A}^\dagger), r^\dagger)} \gg \frac{N_p(\mathcal{L}(\mathbf{A}), r)}{N_p(\mathcal{L}(\mathbf{A}), r; \mathbf{t})} \approx 2^{Cn}$$

(Our hard CVP instance from before is basically just
$\mathbf{A} = I_n$ and $\mathbf{t} = (1/2, \ldots, 1/2)$.)

$$N_p(\widehat{\mathcal{L}}, (r^p + (r^\dagger)^p)^{1/p}) \approx N_p(\mathcal{L}(\mathbf{A}), r) \cdot N_p(\mathcal{L}(\mathbf{A}^\dagger), r^\dagger)$$

$$N_p(\widehat{\mathcal{L}}, (r^p + (r^\dagger)^p)^{1/p}; \widehat{\mathbf{t}}) \approx N_p(\mathcal{L}(\mathbf{A}), r; \mathbf{t}) \cdot N_p(\mathcal{L}(\mathbf{A}^\dagger), r^\dagger; \mathbf{t}^\dagger)$$

# Increasing the Ratio of Close Vectors to Short Vectors

In order to get "many more close vectors than short vectors", we want

$$\frac{N_p(\mathcal{L}(\mathbf{A}^\dagger), r^\dagger; \mathbf{t}^\dagger)}{N_p(\mathcal{L}(\mathbf{A}^\dagger), r^\dagger)} \gg \frac{N_p(\mathcal{L}(\mathbf{A}), r)}{N_p(\mathcal{L}(\mathbf{A}), r; \mathbf{t})} \approx 2^{Cn}$$

(Our hard CVP instance from before is basically just $\mathbf{A} = I_n$ and $\mathbf{t} = (1/2, \ldots, 1/2)$.)

All hardness reductions for SVP use some gadget like this. We show that any such gadget implies hardness.

# Increasing the Ratio of Close Vectors to Short Vectors

$$\widehat{\mathcal{L}} := \mathcal{L}\begin{pmatrix} \mathbf{A} & 0 \\ 0 & \mathbf{A}^\dagger \end{pmatrix}$$

$$\widehat{\mathbf{t}} := \begin{pmatrix} \mathbf{t} \\ \mathbf{t}^\dagger \end{pmatrix}$$

# Increasing the Ratio of Close Vectors to Short Vectors

$$\widehat{\mathcal{L}} := \mathcal{L}\begin{pmatrix} \mathbf{A} & 0 \\ 0 & \mathbf{A}^{\dagger} \end{pmatrix} \qquad \widehat{\mathbf{t}} := \begin{pmatrix} \mathbf{t} \\ \mathbf{t}^{\dagger} \end{pmatrix}$$

$$\mathrm{rank}(\widehat{\mathcal{L}}) = n + n^{\dagger}$$

# Increasing the Ratio of Close Vectors to Short Vectors

$$\widehat{\mathcal{L}} := \mathcal{L}\begin{pmatrix} \mathbf{A} & 0 \\ 0 & \mathbf{A}^\dagger \end{pmatrix} \qquad \widehat{\mathbf{t}} := \begin{pmatrix} \mathbf{t} \\ \mathbf{t}^\dagger \end{pmatrix}$$

$$\mathrm{rank}(\widehat{\mathcal{L}}) = n + n^\dagger$$

$$\frac{N_p(\mathcal{L}(\mathbf{A}^\dagger), r^\dagger; \mathbf{t}^\dagger)}{N_p(\mathcal{L}(\mathbf{A}^\dagger), r^\dagger)} \geq 2^{\Omega(n)}$$

# Increasing the Ratio of Close Vectors to Short Vectors

$$\widehat{\mathcal{L}} := \mathcal{L}\begin{pmatrix} \mathbf{A} & 0 \\ 0 & \mathbf{A}^\dagger \end{pmatrix} \qquad \widehat{\mathbf{t}} := \begin{pmatrix} \mathbf{t} \\ \mathbf{t}^\dagger \end{pmatrix}$$

$$\mathrm{rank}(\widehat{\mathcal{L}}) = n + n^\dagger$$

$$\frac{N_p(\mathcal{L}(\mathbf{A}^\dagger), r^\dagger; \mathbf{t}^\dagger)}{N_p(\mathcal{L}(\mathbf{A}^\dagger), r^\dagger)} \geq 2^{\Omega(n)}$$

To prove $2^{\Omega(n)}$-hardness, we need $n^\dagger = O(n)$. I.e.,

$$. \quad \frac{N_p(\mathcal{L}(\mathbf{A}^\dagger), r^\dagger; \mathbf{t}^\dagger)}{N_p(\mathcal{L}(\mathbf{A}^\dagger), r^\dagger)} \geq 2^{\Omega(n^\dagger)} .$$

# Building the Gadget
# p > 2.14

$$\boxed{\frac{N_p(\mathcal{L}^\dagger, r^\dagger; \mathbf{t}^\dagger)}{N_p(\mathcal{L}^\dagger, r^\dagger)} \geq 2^{\Omega(n^\dagger)}}$$

# Building the Gadget
# p > 2.14

$$\boxed{\frac{N_p(\mathcal{L}^\dagger, r^\dagger; \mathbf{t}^\dagger)}{N_p(\mathcal{L}^\dagger, r^\dagger)} \geq 2^{\Omega(n^\dagger)}}$$

For $p \gtrsim 2.14$, $\mathcal{L}^\dagger := \mathbb{Z}^{n^\dagger}$, $\mathbf{t}^\dagger := (1/2, \ldots, 1/2)$, and $r^\dagger := (n^\dagger)^{1/p}/2$ works!

# Building the Gadget
# p > 2.14

$$\boxed{\frac{N_p(\mathcal{L}^\dagger, r^\dagger; \mathbf{t}^\dagger)}{N_p(\mathcal{L}^\dagger, r^\dagger)} \geq 2^{\Omega(n^\dagger)}}$$

For $p \gtrsim 2.14$, $\mathcal{L}^\dagger := \mathbb{Z}^{n^\dagger}$, $\mathbf{t}^\dagger := (1/2, \ldots, 1/2)$, and $r^\dagger := (n^\dagger)^{1/p}/2$ works!

This is a very convenient gadget!

$r^\dagger := \mathrm{dist}_p(\mathbf{t}^\dagger, \mathcal{L}^\dagger)$ $\quad \widehat{\mathcal{L}} \approx \mathbb{Z}^{n+n^\dagger}$ $\quad \widehat{\mathbf{t}} \approx (1/2, \ldots, 1/2)$

# Building the Gadget p > 2.14

$$\boxed{\frac{N_p(\mathcal{L}^\dagger, r^\dagger; \mathbf{t}^\dagger)}{N_p(\mathcal{L}^\dagger, r^\dagger)} \geq 2^{\Omega(n^\dagger)}}$$

For $p \gtrsim 2.14$, $\mathcal{L}^\dagger := \mathbb{Z}^{n^\dagger}$, $\mathbf{t}^\dagger := (1/2, \ldots, 1/2)$, and $r^\dagger := (n^\dagger)^{1/p}/2$ works!

This is a very convenient gadget!

$r^\dagger := \mathrm{dist}_p(\mathbf{t}^\dagger, \mathcal{L}^\dagger) \qquad \widehat{\mathcal{L}} \approx \mathbb{Z}^{n+n^\dagger} \qquad \widehat{\mathbf{t}} \approx (1/2, \ldots, 1/2)$

We just need to study the number of integer vectors in $\ell_p$ balls.

# Integer points in ℓ_p balls

# Integer points in ℓ_p balls

$$\Theta_p(\tau) := \sum_{z \in \mathbb{Z}} \exp(-\tau |z|^p)$$

# Integer points in ℓ_p balls

$$\Theta_p(\tau) := \sum_{z \in \mathbb{Z}} \exp(-\tau |z|^p)$$

$$\Theta_p(\tau)^n = \sum_{\mathbf{z} \in \mathbb{Z}^n} \exp(-\tau \|\mathbf{z}\|^p)$$

# Integer points in ℓ_p balls

$$\Theta_p(\tau) := \sum_{z \in \mathbb{Z}} \exp(-\tau |z|^p)$$

$$\Theta_p(\tau)^n = \sum_{\mathbf{z} \in \mathbb{Z}^n} \exp(-\tau \|\mathbf{z}\|^p)$$

$$N_p(\mathbb{Z}^n, r) < \exp(\tau r^p) \cdot \Theta_p(\tau)^n$$

# Integer points in ℓ_p balls

$$\Theta_p(\tau) := \sum_{z \in \mathbb{Z}} \exp(-\tau |z|^p)$$

$$\Theta_p(\tau)^n = \sum_{\mathbf{z} \in \mathbb{Z}^n} \exp(-\tau \|\mathbf{z}\|^p)$$

$$N_p(\mathbb{Z}^n, r) < \exp(\tau r^p) \cdot \Theta_p(\tau)^n$$

$$N_p(\mathbb{Z}^n, r) \approx \inf_{\tau > 0} \exp(\tau r^p) \cdot \Theta_p(\tau)^n$$

# Integer points in ℓ_p balls

$$\Theta_p(\tau) := \sum_{z \in \mathbb{Z}} \exp(-\tau |z|^p)$$

$$\Theta_p(\tau)^n = \sum_{\mathbf{z} \in \mathbb{Z}^n} \exp(-\tau \|\mathbf{z}\|^p)$$

$$N_p(\mathbb{Z}^n, r) < \exp(\tau r^p) \cdot \Theta_p(\tau)^n$$

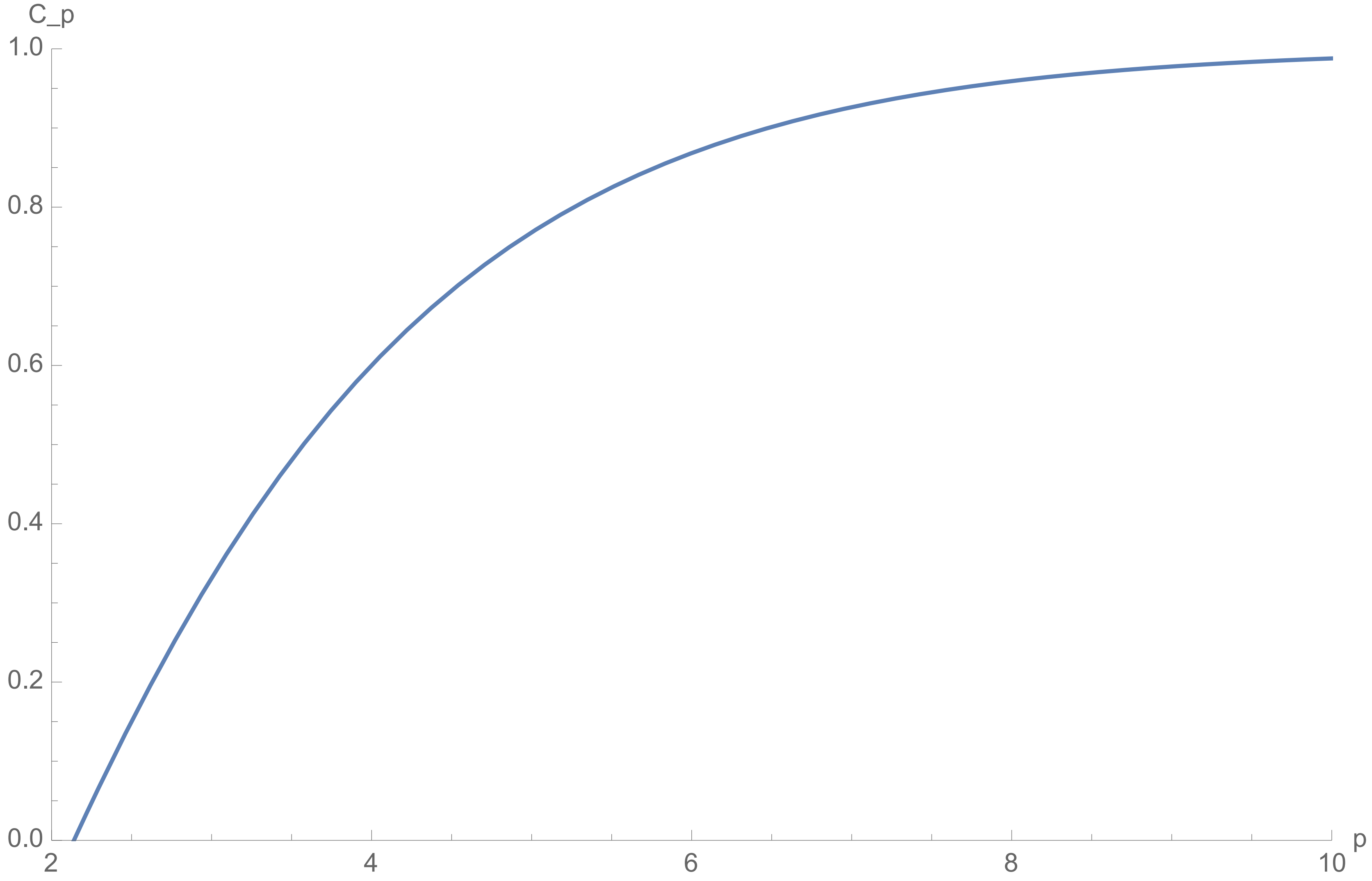
$$N_p(\mathbb{Z}^n, r) \approx \inf_{\tau > 0} \exp(\tau r^p) \cdot \Theta_p(\tau)^n$$

Technique due to [Mazo, Odlyzko 90] and [EOR91].

# SETH-Hardness for p > 2.14 via the Integer Lattice

No $2^{C_p n}$-time algorithm for $\mathsf{SVP}_p$ unless SETH fails.

(For "almost all" $p \gtrsim 2.14$.)



# GapETH-Hardness for p > 2 via the Integer Lattice

# GapETH-Hardness for p > 2 via the Integer Lattice

$\mathcal{L}^\dagger := \mathbb{Z}^{n^\dagger}$ $\qquad$ $\mathbf{t}^\dagger := (t^\dagger, \ldots, t^\dagger)$ $\qquad$ $r^\dagger = \Theta(n^{1/p})$

# GapETH-Hardness for p > 2 via the Integer Lattice

$$\mathcal{L}^\dagger := \mathbb{Z}^{n^\dagger} \qquad \mathbf{t}^\dagger := (t^\dagger, \ldots, t^\dagger) \qquad r^\dagger = \Theta(n^{1/p})$$

(To analyze this, we study $\Theta_p(\tau; t^\dagger) := \sum_{z \in \mathbb{Z}} e^{-\tau |z - t^\dagger|^p}$.)

# GapETH-Hardness for p > 2 via the Integer Lattice

$\mathcal{L}^\dagger := \mathbb{Z}^{n^\dagger}$ $\mathbf{t}^\dagger := (t^\dagger, \ldots, t^\dagger)$ $r^\dagger = \Theta(n^{1/p})$

(To analyze this, we study $\Theta_p(\tau; t^\dagger) := \sum_{z \in \mathbb{Z}} e^{-\tau|z - t^\dagger|^p}$.)

$2^{\Omega(n)}$-hardness assuming no $2^{o(n)}$-time algorithm for Gap-2-SAT.

# GapETH-Hardness for p > 2 via the Integer Lattice

$\mathcal{L}^\dagger := \mathbb{Z}^{n^\dagger}$ $\qquad$ $\mathbf{t}^\dagger := (t^\dagger, \ldots, t^\dagger)$ $\qquad$ $r^\dagger = \Theta(n^{1/p})$

(To analyze this, we study $\Theta_p(\tau; t^\dagger) := \sum_{z \in \mathbb{Z}} e^{-\tau|z - t^\dagger|^p}$.)

$2^{\Omega(n)}$-hardness assuming no $2^{o(n)}$-time algorithm for Gap-2-SAT.

Because $r^\dagger > \mathrm{dist}(\mathbf{t}^\dagger, \mathcal{L}^\dagger)$, we need to reduce from approx-CVP, so we get weaker hardness.

# GapETH-Hardness for p > 2 via the Integer Lattice

$\mathcal{L}^\dagger := \mathbb{Z}^{n^\dagger}$ $\qquad$ $\mathbf{t}^\dagger := (t^\dagger, \ldots, t^\dagger)$ $\qquad$ $r^\dagger = \Theta(n^{1/p})$

(To analyze this, we study $\Theta_p(\tau; t^\dagger) := \sum_{z \in \mathbb{Z}} e^{-\tau |z - t^\dagger|^p}$.)

$2^{\Omega(n)}$-hardness assuming no $2^{o(n)}$-time algorithm for Gap-2-SAT.

Because $r^\dagger > \mathrm{dist}(\mathbf{t}^\dagger, \mathcal{L}^\dagger)$, we need to reduce from approx-CVP, so we get weaker hardness.

(The integer lattice can't work for $p \leq 2$.)

# What about p = 2?!

$$\boxed{\frac{N_p(\mathcal{L}^\dagger, r^\dagger; \mathbf{t}^\dagger)}{N_p(\mathcal{L}^\dagger, r^\dagger)} \geq 2^{\Omega(n^\dagger)}}$$

# What about p = 2?!

$$\boxed{\frac{N_p(\mathcal{L}^\dagger, r^\dagger; \mathbf{t}^\dagger)}{N_p(\mathcal{L}^\dagger, r^\dagger)} \geq 2^{\Omega(n^\dagger)}}$$

- Implies hardness for p < 2 [Regev, Rosen ’06].

# What about p = 2?!

$$\boxed{\frac{N_p(\mathcal{L}^\dagger, r^\dagger; \mathbf{t}^\dagger)}{N_p(\mathcal{L}^\dagger, r^\dagger)} \geq 2^{\Omega(n^\dagger)}}$$

- Implies hardness for p < 2 [Regev, Rosen ’06].
- Certain (reasonable?) geometric conjectures yield such a gadget.

# What about p = 2?!

$$\boxed{\frac{N_p(\mathcal{L}^\dagger, r^\dagger; \mathbf{t}^\dagger)}{N_p(\mathcal{L}^\dagger, r^\dagger)} \geq 2^{\Omega(n^\dagger)}}$$

- Implies hardness for p < 2 [Regev, Rosen ’06].
- Certain (reasonable?) geometric conjectures yield such a gadget.
- Most natural: lattice with exponential kissing number.

# What about p = 2?!

$$\boxed{\frac{N_p(\mathcal{L}^\dagger, r^\dagger; \mathbf{t}^\dagger)}{N_p(\mathcal{L}^\dagger, r^\dagger)} \geq 2^{\Omega(n^\dagger)}}$$

- Implies hardness for p < 2 [Regev, Rosen '06].
- Certain (reasonable?) geometric conjectures yield such a gadget.
- Most natural: lattice with exponential kissing number.
    - I.e., $N_2(\mathcal{L}^\dagger, \lambda_1^{(2)}(\mathcal{L}^\dagger)) \geq 2^{\Omega(n^\dagger)}$.

# What about p = 2?!

$$\boxed{\frac{N_p(\mathcal{L}^\dagger, r^\dagger; \mathbf{t}^\dagger)}{N_p(\mathcal{L}^\dagger, r^\dagger)} \geq 2^{\Omega(n^\dagger)}}$$

- Implies hardness for p < 2 [Regev, Rosen '06].
- Certain (reasonable?) geometric conjectures yield such a gadget.
- Most natural: lattice with exponential kissing number.
  - I.e., $N_2(\mathcal{L}^\dagger, \lambda_1^{(2)}(\mathcal{L}^\dagger)) \geq 2^{\Omega(n^\dagger)}$.
  - (Just take $r^\dagger = (1-\varepsilon)\lambda_1^{(2)}(\mathcal{L}), \mathbf{t}^\dagger \approx \mathbf{0}$.)

# What about p = 2?!

$$\boxed{\frac{N_p(\mathcal{L}^\dagger, r^\dagger; \mathbf{t}^\dagger)}{N_p(\mathcal{L}^\dagger, r^\dagger)} \geq 2^{\Omega(n^\dagger)}}$$

- Implies hardness for p < 2 [Regev, Rosen ’06].
- Certain (reasonable?) geometric conjectures yield such a gadget.
- Most natural: lattice with exponential kissing number.
    - I.e., $N_2(\mathcal{L}^\dagger, \lambda_1^{(2)}(\mathcal{L}^\dagger)) \geq 2^{\Omega(n^\dagger)}$.
    - (Just take $r^\dagger = (1-\varepsilon)\lambda_1^{(2)}(\mathcal{L}), \mathbf{t}^\dagger \approx \mathbf{0}$.)
    - Seems hard, so we give weaker conditions that also suffice.

# What about p = 2?!

$$\boxed{\frac{N_p(\mathcal{L}^\dagger, r^\dagger; \mathbf{t}^\dagger)}{N_p(\mathcal{L}^\dagger, r^\dagger)} \geq 2^{\Omega(n^\dagger)}}$$

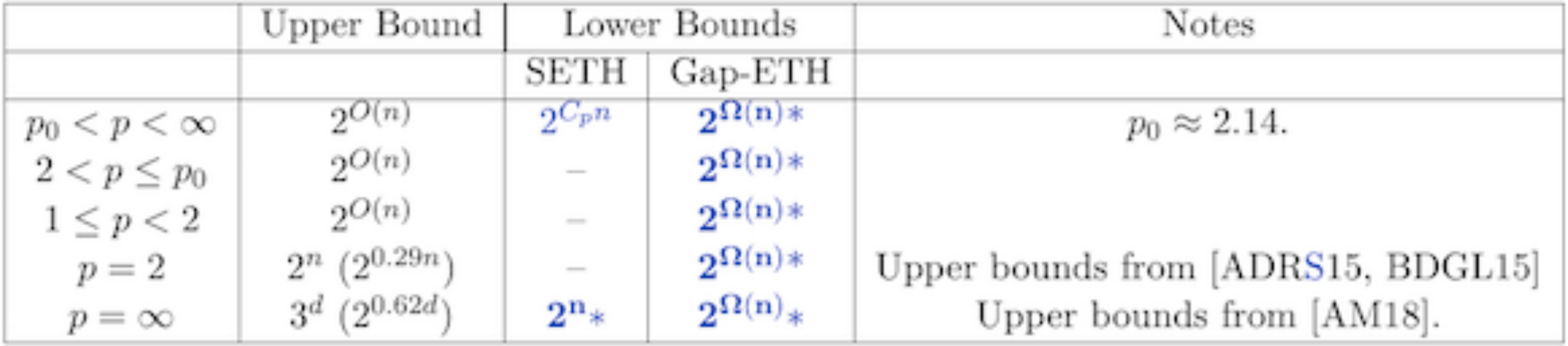
- Implies hardness for p < 2 [Regev, Rosen ’06].
- Certain (reasonable?) geometric conjectures yield such a gadget.
- Most natural: lattice with exponential kissing number.
  - I.e., $N_2(\mathcal{L}^\dagger, \lambda_1^{(2)}(\mathcal{L}^\dagger)) \geq 2^{\Omega(n^\dagger)}$.
  - (Just take $r^\dagger = (1-\varepsilon)\lambda_1^{(2)}(\mathcal{L}), \mathbf{t}^\dagger \approx \mathbf{0}$.)
  - Seems hard, so we give weaker conditions that also suffice.
  - Proven by Serge Vlăduţ in February!!

# What about p = 2?!

SVP$_2$ is $2^{\Omega(n)}$-hard unless GapETH fails!

- Implies hardness for p < 2 [Regev, Rosen '06].
- Certain (reasonable?) geometric conjectures yield such a gadget.
- Most natural: lattice with exponential kissing number.
    - I.e., $N_2(\mathcal{L}^\dagger, \lambda_1^{(2)}(\mathcal{L}^\dagger)) \geq 2^{\Omega(n^\dagger)}$.
    - (Just take $r^\dagger = (1-\varepsilon)\lambda_1^{(2)}(\mathcal{L}), \mathbf{t}^\dagger \approx \mathbf{0}$.)
    - Seems hard, so we give weaker conditions that also suffice.
    - Proven by Serge Vlăduţ in February!!

# Summary

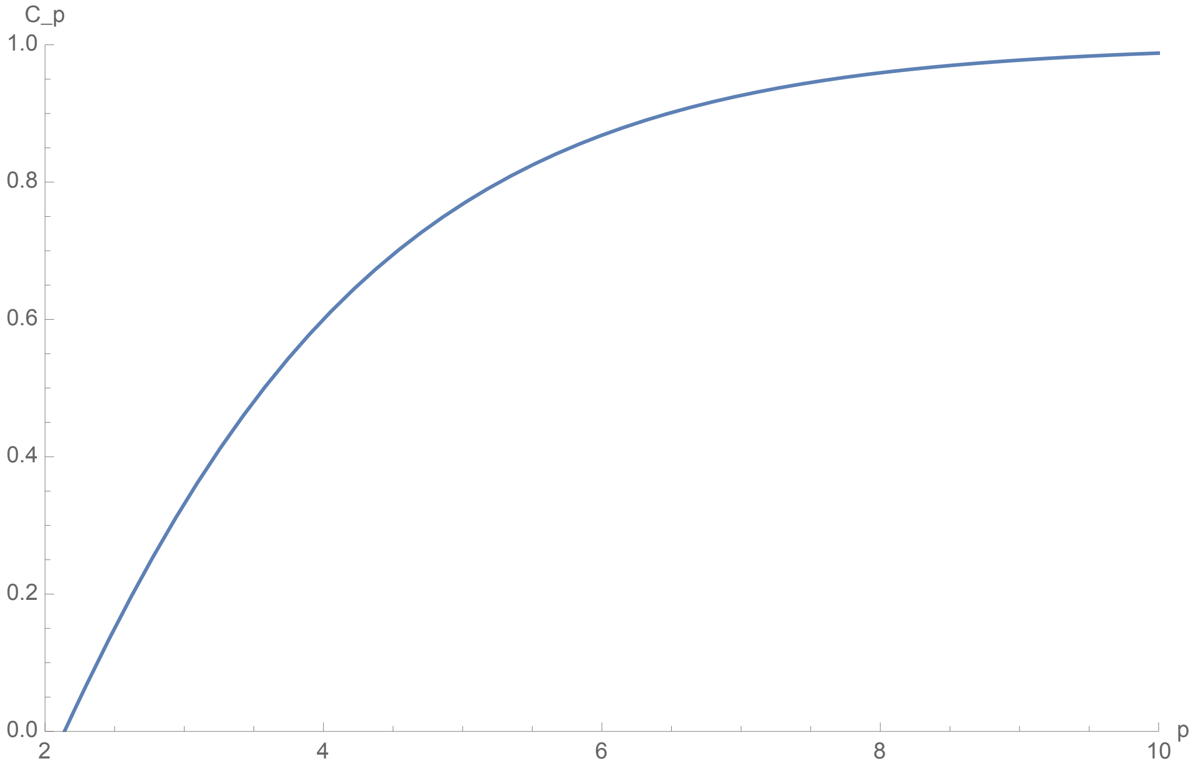


| | Upper Bound | Lower Bounds | | Notes |
|---|---|---|---|---|
| | | SETH | Gap-ETH | |
| $p_0 < p < \infty$ | $2^{O(n)}$ | $2^{C_p n}$ | $2^{\Omega(n)}*$ | $p_0 \approx 2.14$. |
| $2 < p \le p_0$ | $2^{O(n)}$ | – | $2^{\Omega(n)}*$ | |
| $1 \le p < 2$ | $2^{O(n)}$ | – | $2^{\Omega(n)}*$ | |
| $p = 2$ | $2^n$ $(2^{0.29n})$ | – | $2^{\Omega(n)}*$ | Upper bounds from [ADRS15, BDGL15] |
| $p = \infty$ | $3^d$ $(2^{0.62d})$ | $2^n*$ | $2^{\Omega(n)}*$ | Upper bounds from [AM18]. |

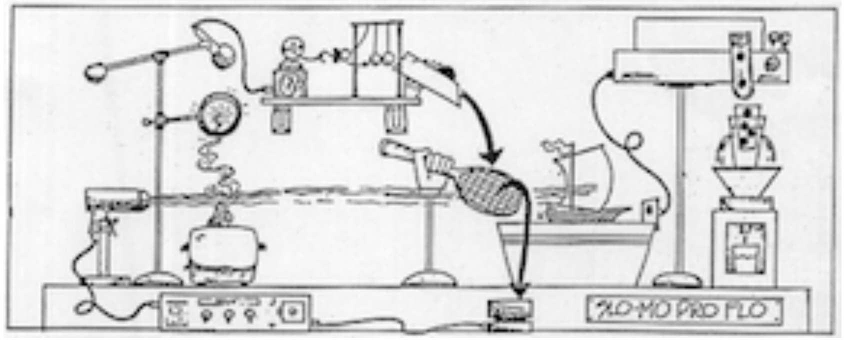
Blue = new result.

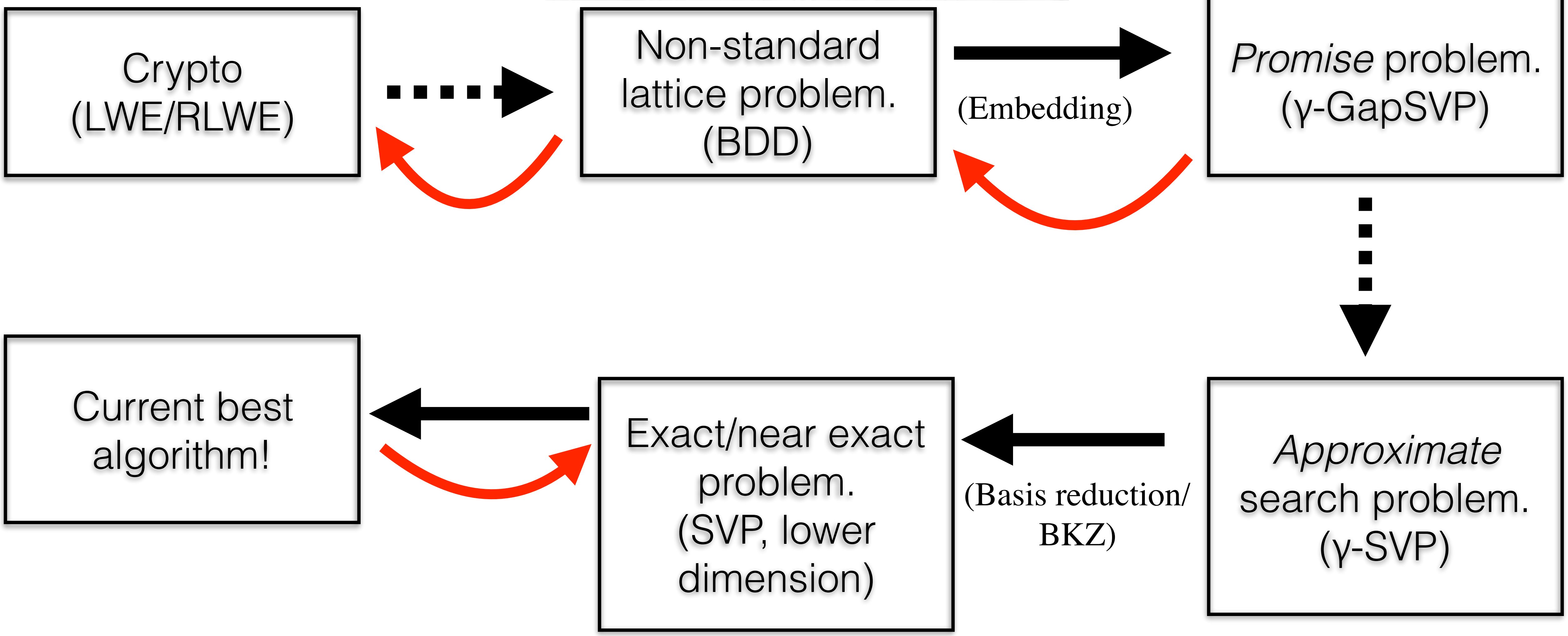
(…) = heuristic algorithm

* = hardness for some constant approximation factor



# The Path Forward





Crypto (LWE/RLWE)

Non-standard lattice problem. (BDD)

(Embedding)

*Promise* problem. (γ-GapSVP)

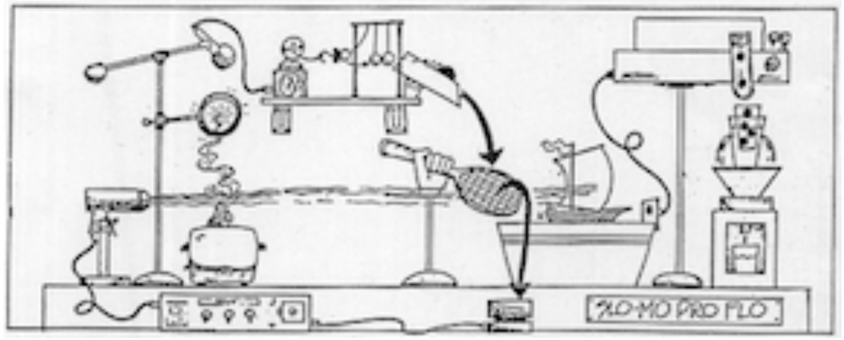
*Approximate* search problem. (γ-SVP)

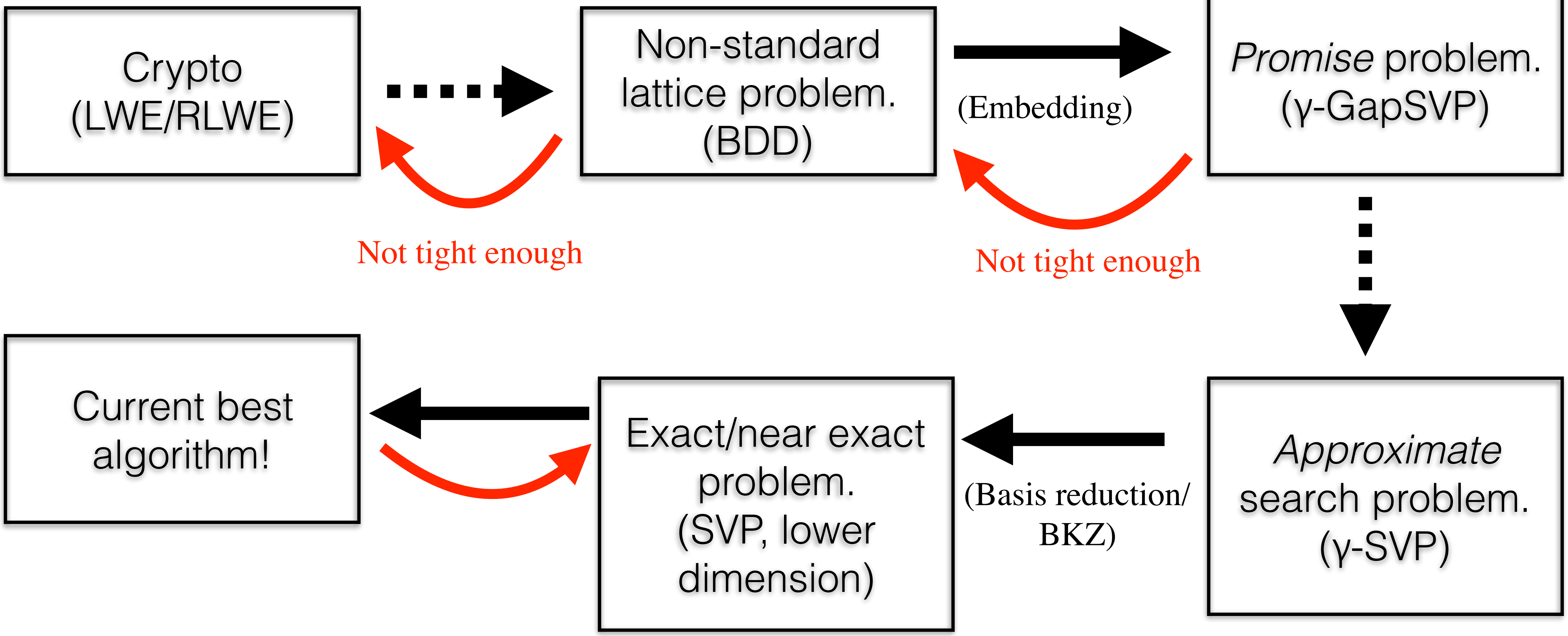
(Basis reduction/ BKZ)

Exact/near exact problem. (SVP, lower dimension)

Current best algorithm!

# The Path Forward





Crypto (LWE/RLWE)

Non-standard lattice problem. (BDD)

(Embedding)

*Promise* problem. (γ-GapSVP)

Not tight enough

Not tight enough

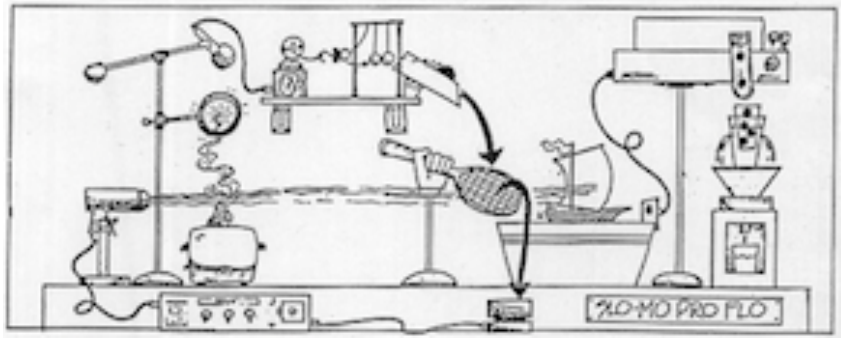
Current best algorithm!

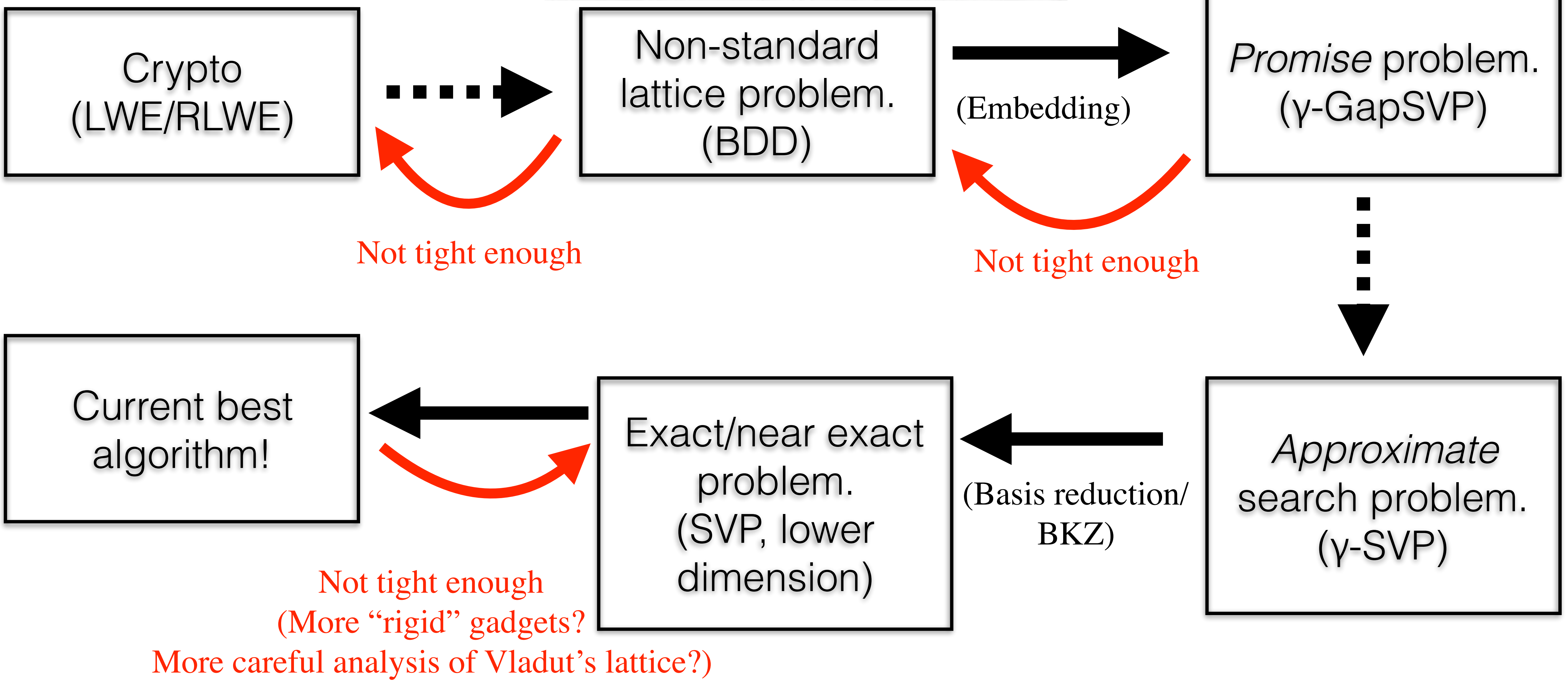
Exact/near exact problem. (SVP, lower dimension)

(Basis reduction/ BKZ)

*Approximate* search problem. (γ-SVP)

# The Path Forward





Crypto (LWE/RLWE)

Non-standard lattice problem. (BDD)

(Embedding)

*Promise* problem. (γ-GapSVP)

Not tight enough

Not tight enough

Current best algorithm!

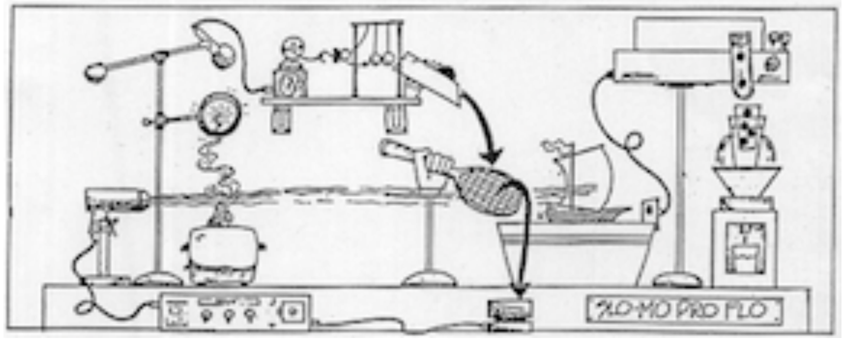
Exact/near exact problem. (SVP, lower dimension)

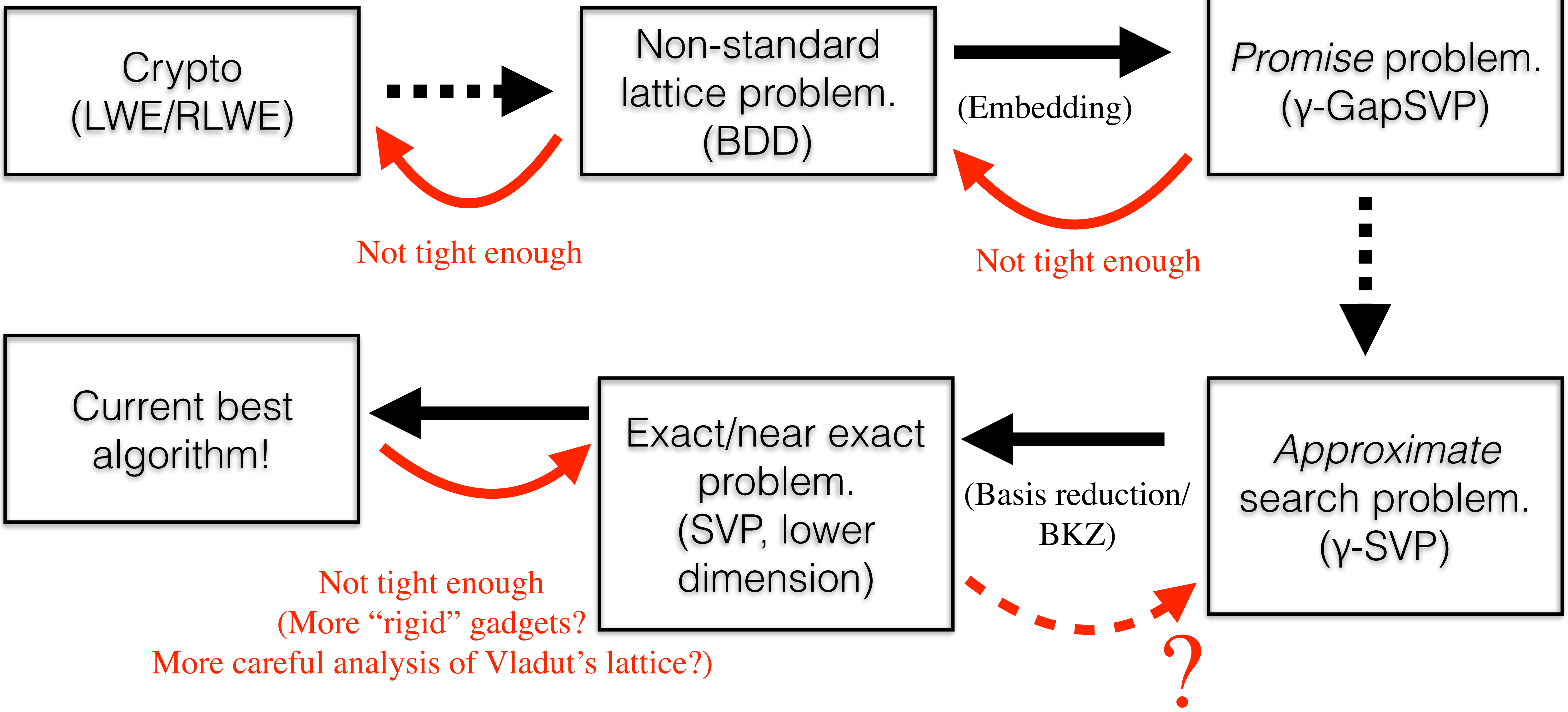
(Basis reduction/ BKZ)

*Approximate* search problem. (γ-SVP)

Not tight enough
(More “rigid” gadgets?
More careful analysis of Vladut’s lattice?)

# The Path Forward





Crypto (LWE/RLWE)

Non-standard lattice problem. (BDD)

(Embedding)

*Promise* problem. (γ-GapSVP)

Not tight enough

Not tight enough

Current best algorithm!

Exact/near exact problem. (SVP, lower dimension)

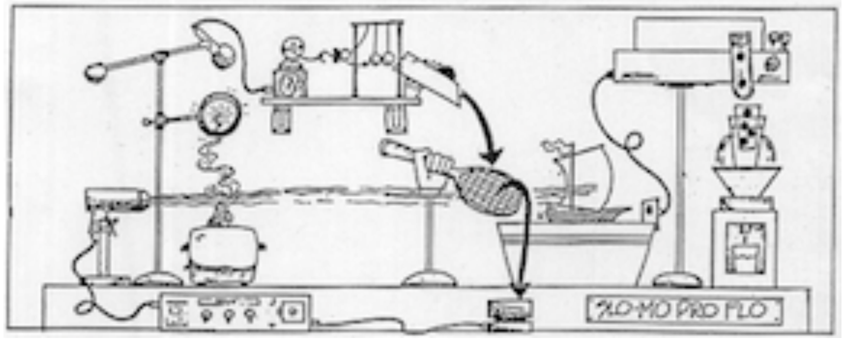
(Basis reduction/ BKZ)

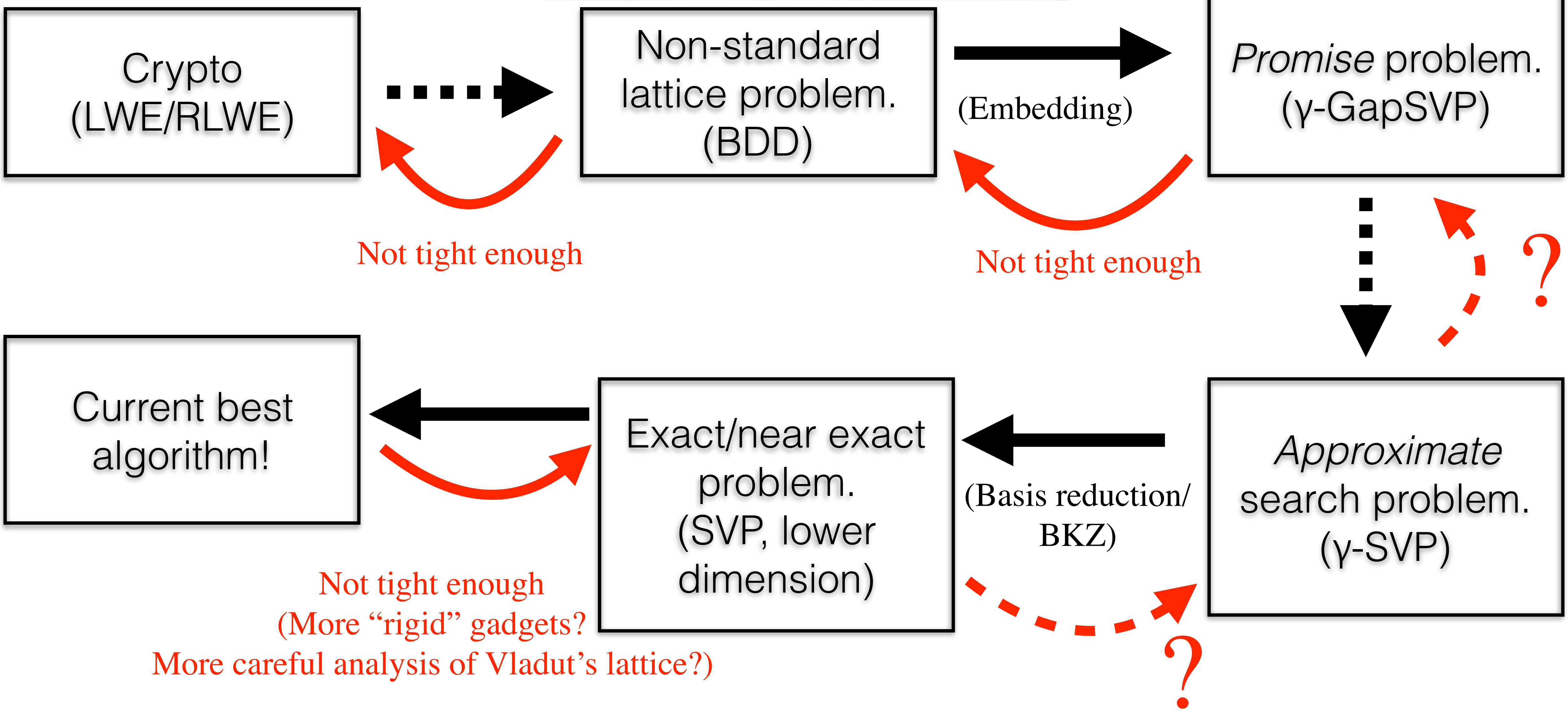
*Approximate* search problem. (γ-SVP)

Not tight enough
(More "rigid" gadgets?
More careful analysis of Vladut's lattice?)

?

# The Path Forward





Crypto (LWE/RLWE)

Non-standard lattice problem. (BDD)

(Embedding)

*Promise* problem. (γ-GapSVP)

Not tight enough

Not tight enough

?

*Approximate* search problem. (γ-SVP)

(Basis reduction/ BKZ)

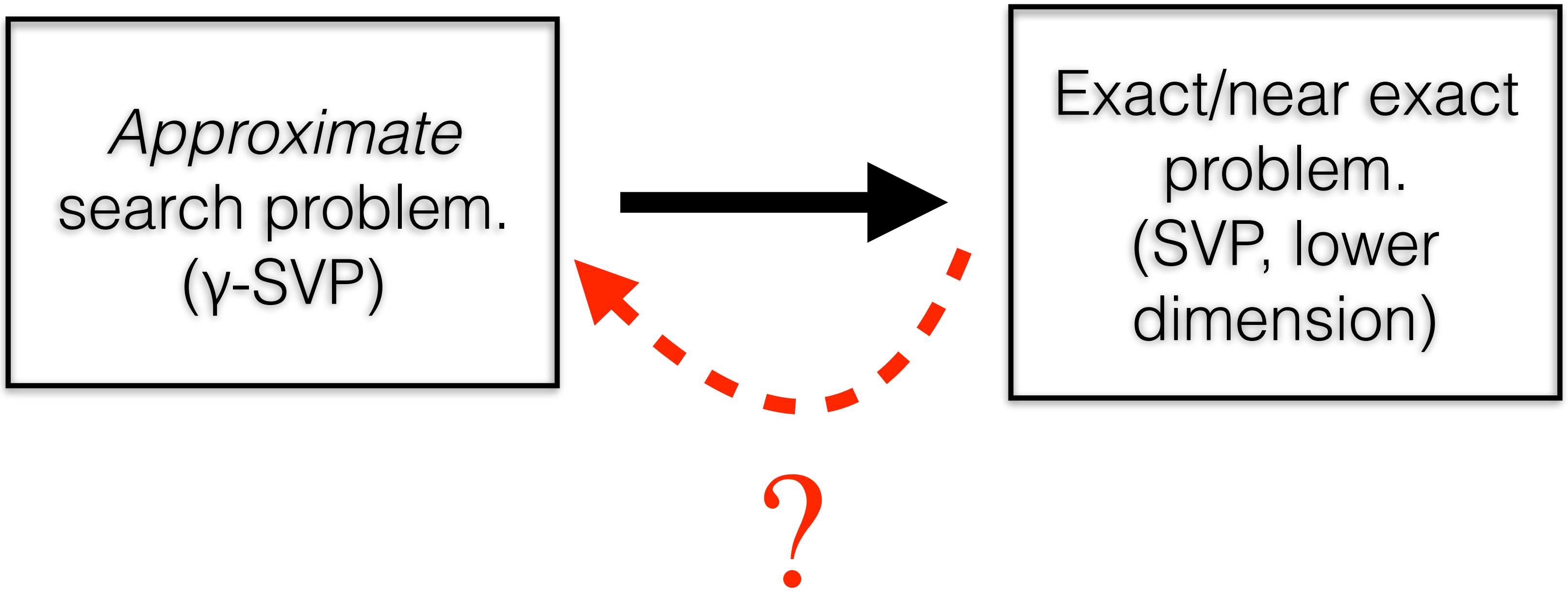
?

Exact/near exact problem. (SVP, lower dimension)

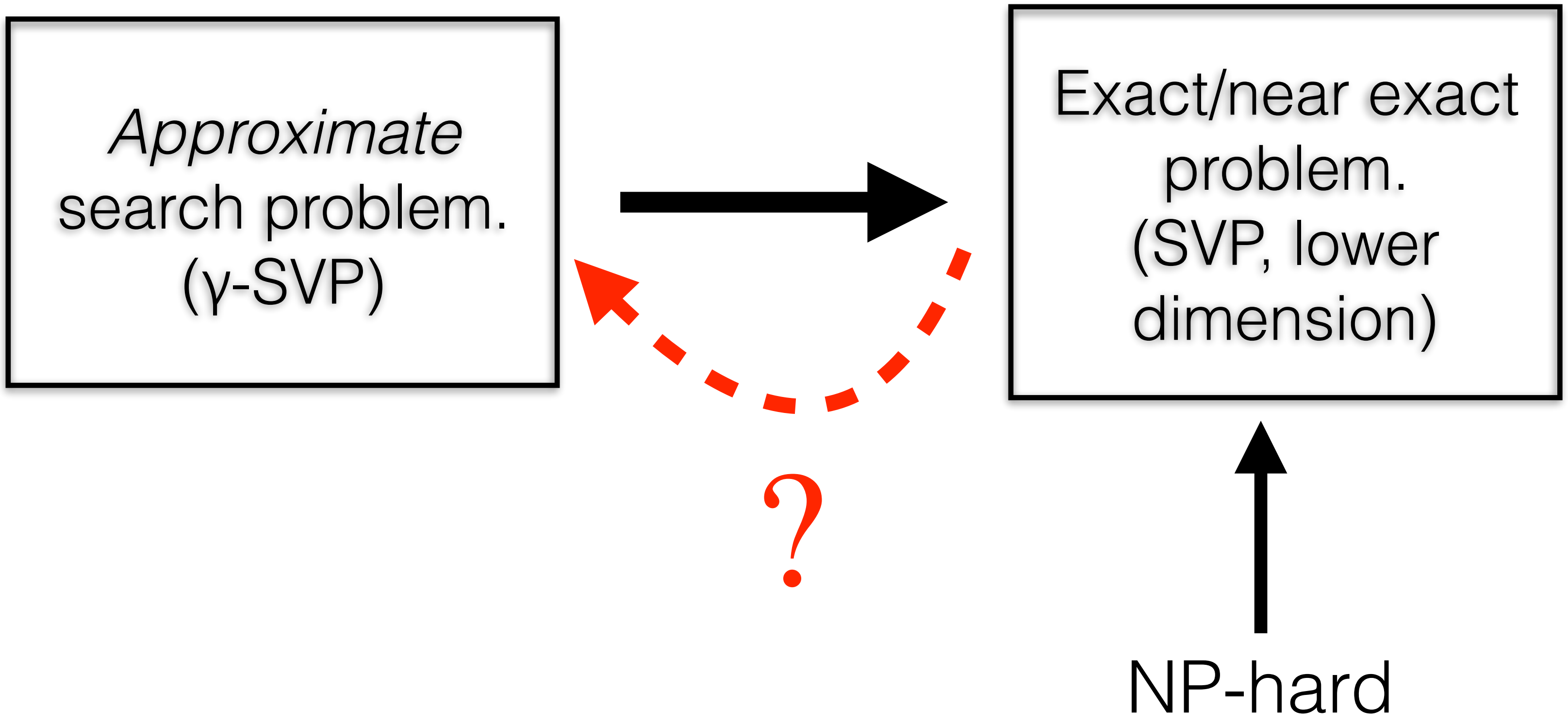
Current best algorithm!

Not tight enough
(More "rigid" gadgets?
More careful analysis of Vladut's lattice?)

# The Path Forward—
# Is BKZ Optimal?


*Approximate* search problem. (γ-SVP)
Exact/near exact problem. (SVP, lower dimension)
?

# The Path Forward—Is BKZ Optimal?



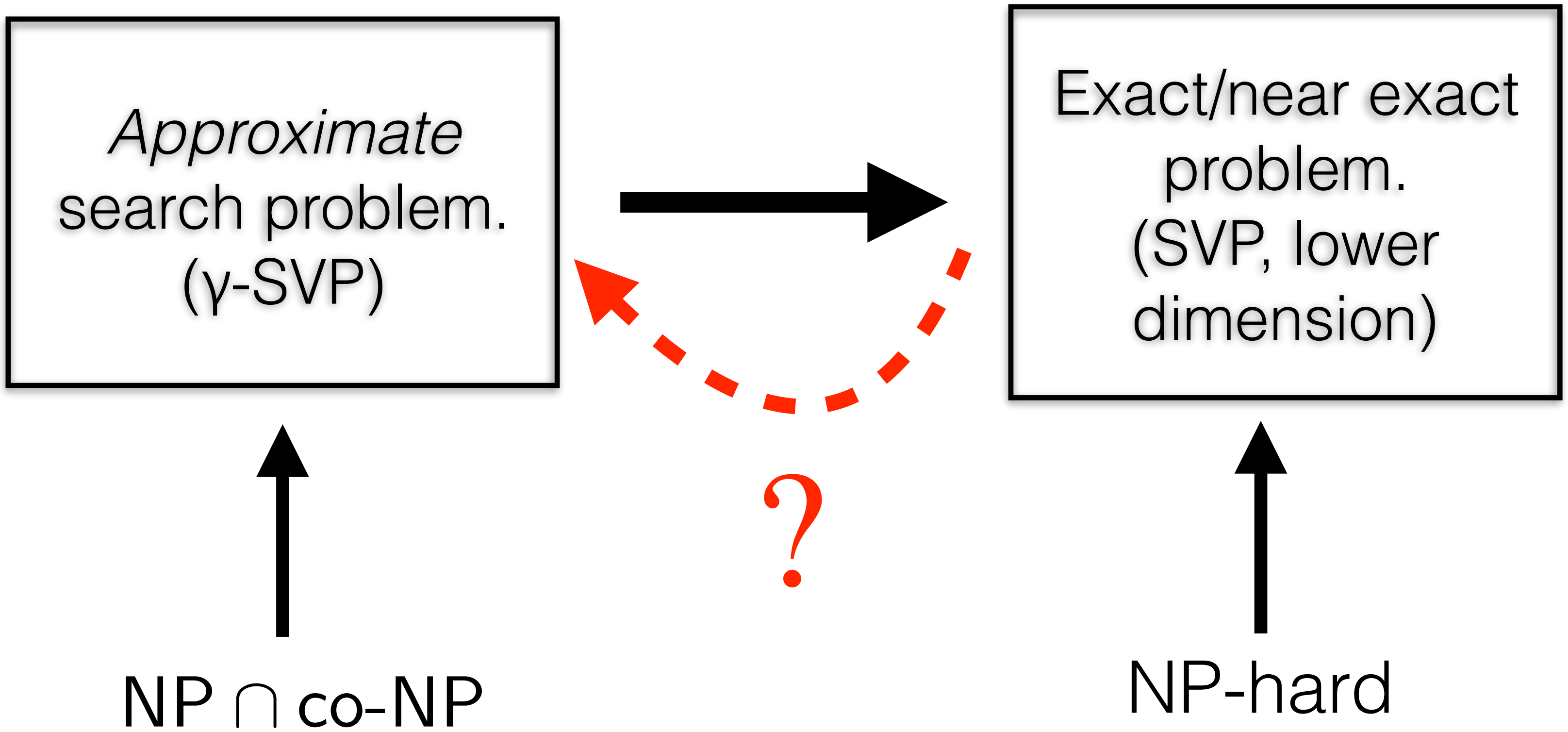
*Approximate* search problem. (γ-SVP)

Exact/near exact problem. (SVP, lower dimension)

?

NP-hard

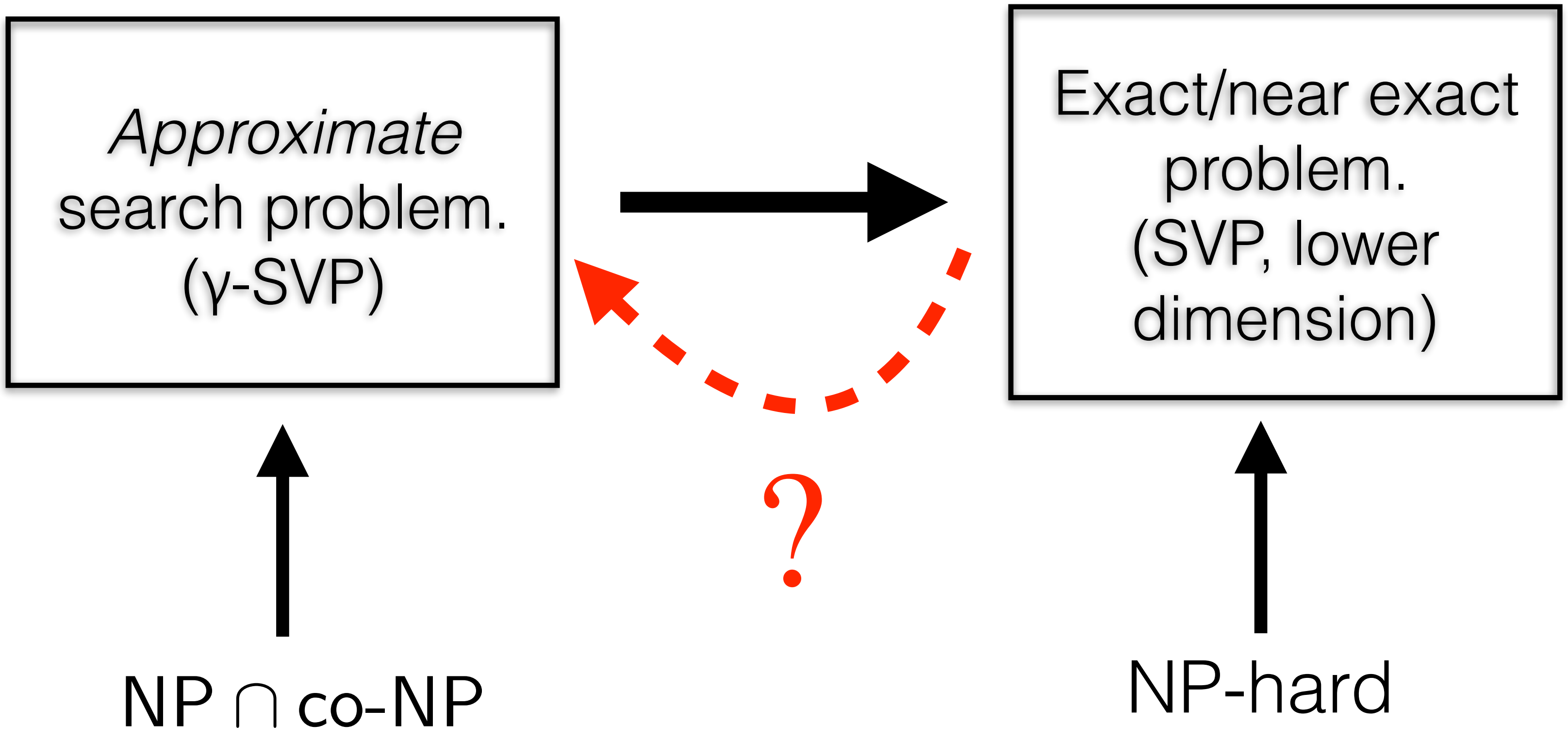
# The Path Forward—
# Is BKZ Optimal?



*Approximate* search problem. (γ-SVP)

Exact/near exact problem. (SVP, lower dimension)

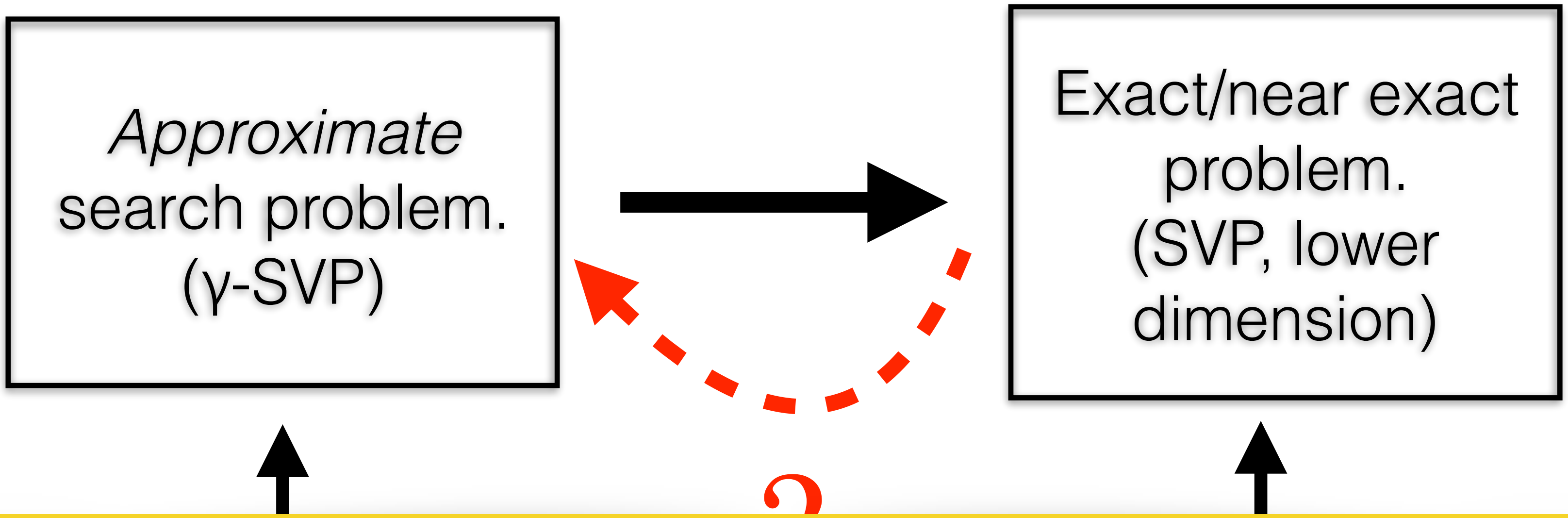
# The Path Forward—
# Is BKZ Optimal?


*Approximate* search problem. (γ-SVP)
Exact/near exact problem. (SVP, lower dimension)
?
NP ∩ co-NP
NP-hard

Any reduction in the other direction has to be “interesting.”

Superpolynomial? Non-deterministic? Non-uniform?

# The Path Forward—
# Is BKZ Optimal?



*Approximate* search problem. (γ-SVP)

Exact/near exact problem. (SVP, lower dimension)

?

Maybe BKZ is fundamentally the wrong approach for approximate lattice problems?

Any reduction in the other direction has to be “interesting.”

Superpolynomial? Non-deterministic? Non-uniform?

# Thanks!