How to Find Ternary LWE Keys Using Locality Sensitive Hashing

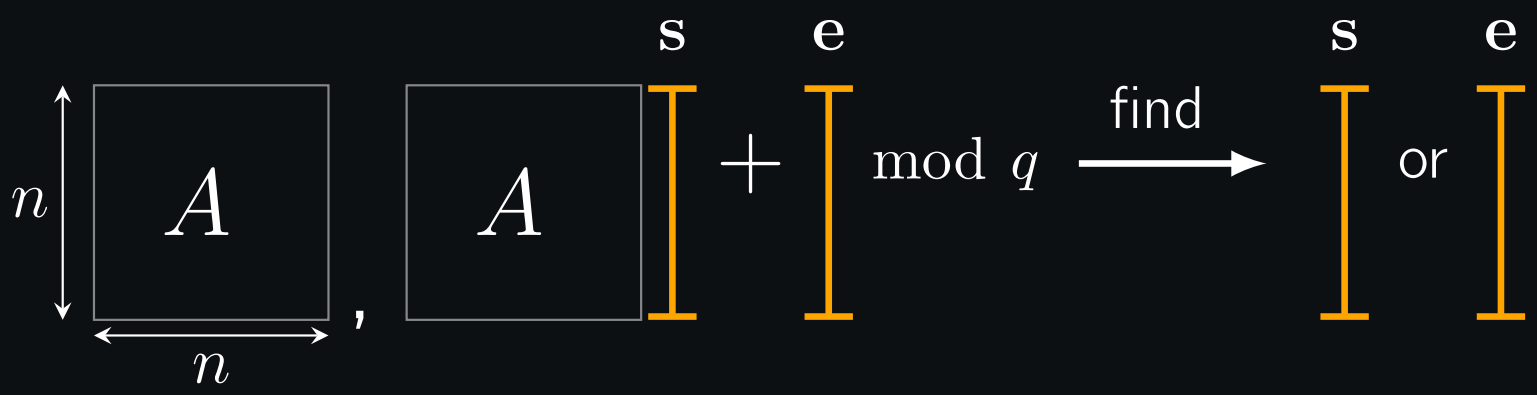
Elena Kirshanova, Alexander May

$$\sqrt{\tfrac{3}{4}}\,|\text{I. Kant BFU}\rangle + \sqrt{\tfrac{1}{4}}\,|\text{RUB}\rangle$$

18th IMA International Conference On Cryptography and Coding

December 15, 2021

Ternary LWE



$$A \xleftarrow{\$} \mathbb{Z}_q^{n \times n} \qquad \mathbf{s}, \mathbf{e} \xleftarrow{\$} \{-1, 0, 1\}^n$$

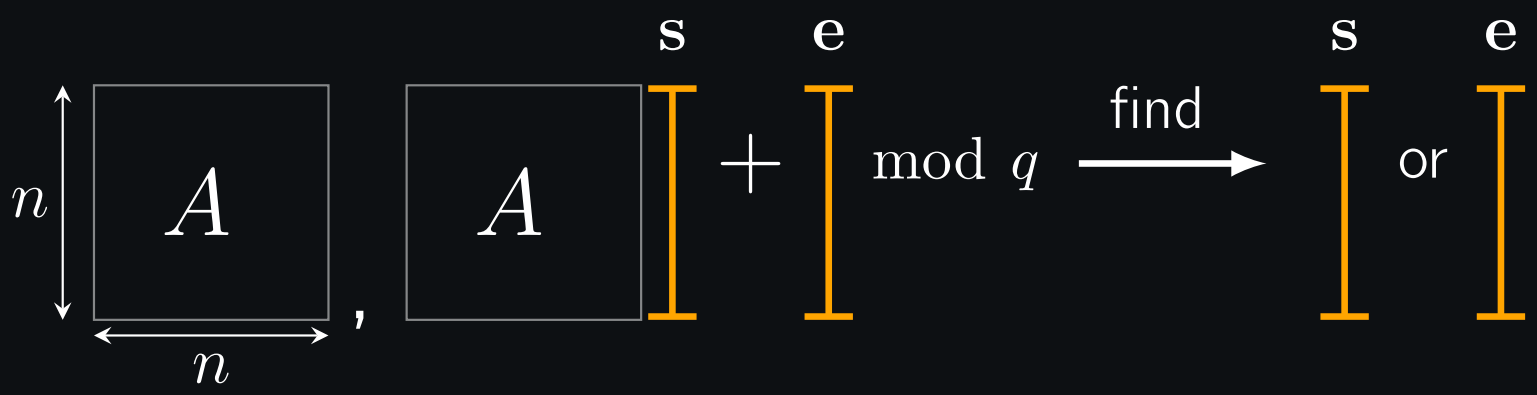
Relevant to NTRU-types schemes[1] and some signatures.[2]

---

[1] J. Hoffstein, J. Pipher, J. H. Silverman. Ntru: A ring-based public key cryptosystem. ANTS'98

[2] L. Ducas, A. Durmus, T. Lepoint, V. Lyubashevsky. Lattice signatures and bimodal Gaussians. Crypto'13

Ternary LWE



$$A \xleftarrow{\$} \mathbb{Z}_q^{n \times n} \qquad \mathbf{s}, \mathbf{e} \xleftarrow{\$} \{-1, 0, 1\}^n$$

Relevant to NTRU-types schemes[1] and some signatures.[2]

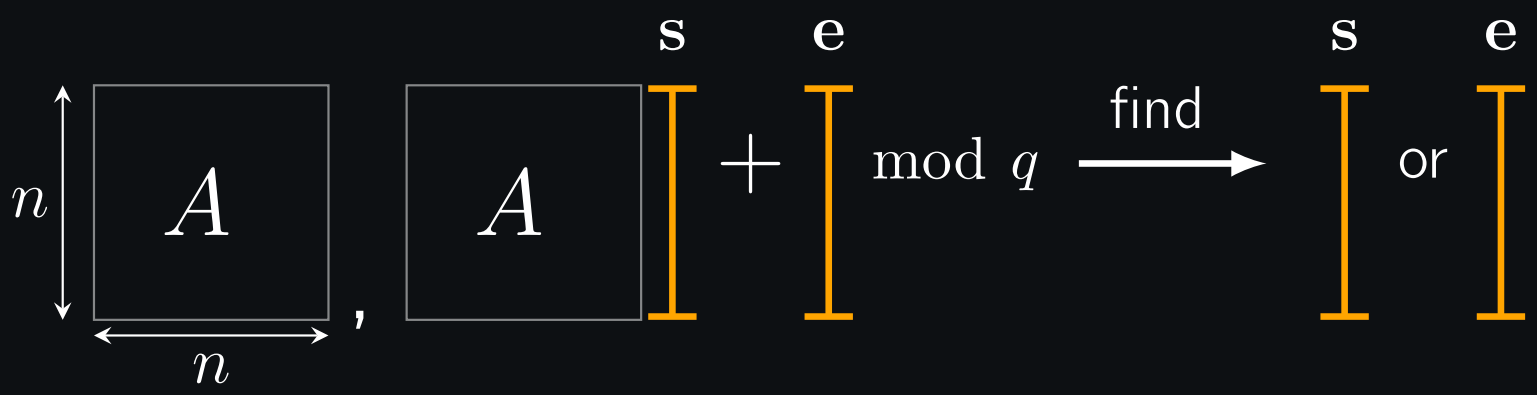
Main approaches to solve ternary LWE:

- lattice basis reduction attacks
- combinatorial attacks

[1] J. Hoffstein, J. Pipher, J. H. Silverman. Ntru: A ring-based public key cryptosystem. ANTS'98

[2] L. Ducas, A. Durmus, T. Lepoint, V. Lyubashevsky. Lattice signatures and bimodal Gaussians. Crypto'13

Ternary LWE



$$A \xleftarrow{\$} \mathbb{Z}_q^{n \times n} \qquad \mathbf{s}, \mathbf{e} \xleftarrow{\$} \{-1, 0, 1\}^n$$

Relevant to NTRU-types schemes[1] and some signatures.[2]

Main approaches to solve ternary LWE:

- lattice basis reduction attacks
- combinatorial attacks. This talk

[1] J. Hoffstein, J. Pipher, J. H. Silverman. Ntru: A ring-based public key cryptosystem. ANTS'98

[2] L. Ducas, A. Durmus, T. Lepoint, V. Lyubashevsky. Lattice signatures and bimodal Gaussians. Crypto'13

Prior art I.Oldyzko's idea

$$A\mathbf{s} + \mathbf{e} = \mathbf{b} \bmod q$$

$$\Longleftrightarrow$$

$$A\mathbf{s}_1 \approx_{\mathbf{e}} A\mathbf{s}_2 + \mathbf{b} \bmod q$$

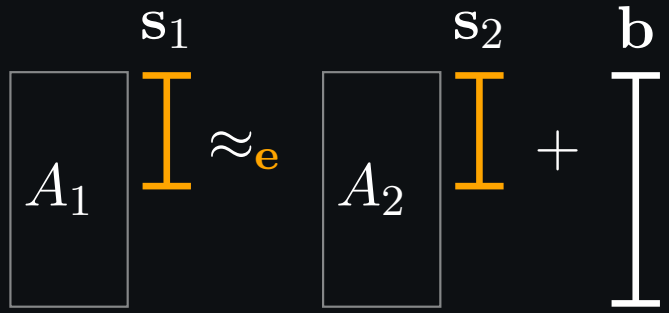
Prior art I.Oldyzko's idea

$$A\mathbf{s} + \mathbf{e} = \mathbf{b} \bmod q$$

$$\Longleftrightarrow$$

$$A\mathbf{s}_1 \approx_{\mathbf{e}} A\mathbf{s}_2 + \mathbf{b} \bmod q$$

Oldyzko'98: $A = (A_1|A_2) \in \mathbb{Z}_q^{n \times n}$



- $L_1 = \{\mathbf{s}_1 \in \{\pm 1, 0\}^{\frac{n}{2}}, \mathrm{Hash}(A_1\mathbf{s}_1)\}$
- $L_2 = \{\mathbf{s}_2 \in \{\pm 1, 0\}^{\frac{n}{2}}, \mathrm{Hash}(-A_2\mathbf{s}_2 + \mathbf{b})\}$
- Sort L_1, L_2 w.r.t. $\mathrm{Hash}()$
- Output $\mathbf{s}_1, \mathbf{s}_2$ s.t. $\mathrm{Hash}(A\mathbf{s}_1) = \mathrm{Hash}(-A\mathbf{s}_2 + \mathbf{b})$

Time $\approx$ Space $\approx (\underbrace{3^n}_{\text{search space}})^{1/2}$

Prior art I.Oldyzko's idea

$$A\mathbf{s} + \mathbf{e} = \mathbf{b} \bmod q$$

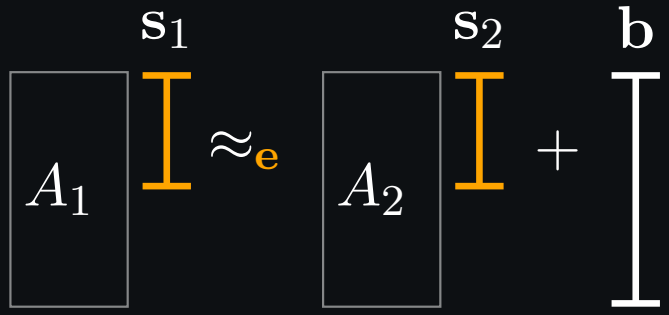
$$\Longleftrightarrow$$

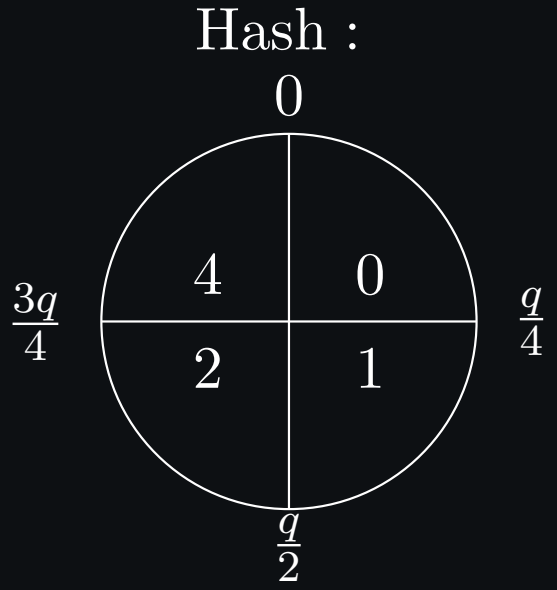
$$A\mathbf{s}_1 \approx_{\mathbf{e}} A\mathbf{s}_2 + \mathbf{b} \bmod q$$

Oldyzko'98: $A = (A_1|A_2) \in \mathbb{Z}_q^{n \times n}$

- $L_1 = \{\mathbf{s}_1 \in \{\pm 1, 0\}^{\frac{n}{2}}, \mathrm{Hash}(A_1\mathbf{s}_1)\}$
- $L_2 = \{\mathbf{s}_2 \in \{\pm 1, 0\}^{\frac{n}{2}}, \mathrm{Hash}(-A_2\mathbf{s}_2 + \mathbf{b})\}$
- Sort L_1, L_2 w.r.t. $\mathrm{Hash}()$
- Output $\mathbf{s}_1, \mathbf{s}_2$ s.t. $\mathrm{Hash}(A\mathbf{s}_1) = \mathrm{Hash}(-A\mathbf{s}_2 + \mathbf{b})$

Time $\approx$ Space $\approx$ ($\underbrace{3^n}_{\text{search space}}$)$^{1/2}$





Prior art II. Representation technique

$$A\mathbf{s} + \mathbf{e} = \mathbf{b} \bmod q$$
$$\Longleftrightarrow$$
$$A\mathbf{s}_1 \approx_{\mathbf{e}} A\mathbf{s}_2 + \mathbf{b} \bmod q$$

May'21: $A \in \mathbb{Z}_q^{n \times n}$, $\omega = |\{i : \mathbf{s}_i \neq 0\}|$ - weight of $\mathbf{s}$, #1's = #-1's

Idea: If we enumerate $\mathbf{s}_1, \mathbf{s}_2 \in \{\pm 1, 0\}^n$, we have

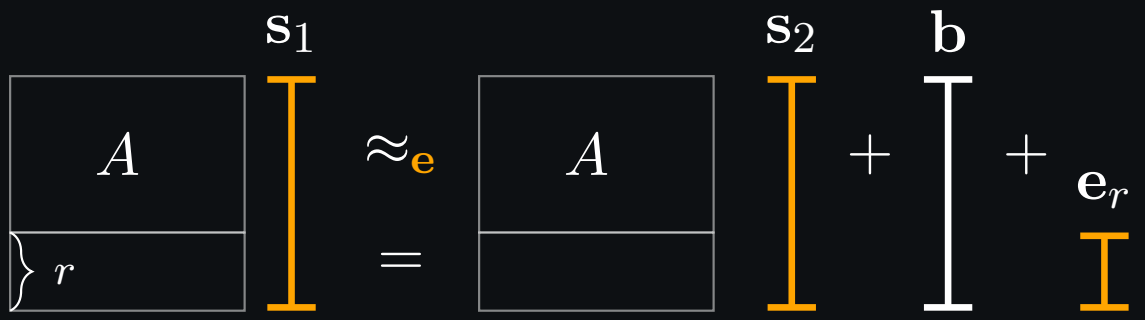
$$R = \underbrace{\binom{\omega/2}{\omega/4}}_{\text{\#reps. for '1's}} \cdot \underbrace{\binom{\omega/2}{\omega/4}}_{\text{\#reps. for '-1's}}$$

ways to represent $\mathbf{s} = \mathbf{s}_1 + \mathbf{s}_2$. We may search for 'special' $\mathbf{s}_1, \mathbf{s}_2$.

This is known as the representation technique[3]

[3] A. Becker, J-S. Coron, A. Joux. Improved generic algorithms for hard knapsacks. Eurocrypt'11

Prior art II. May's idea

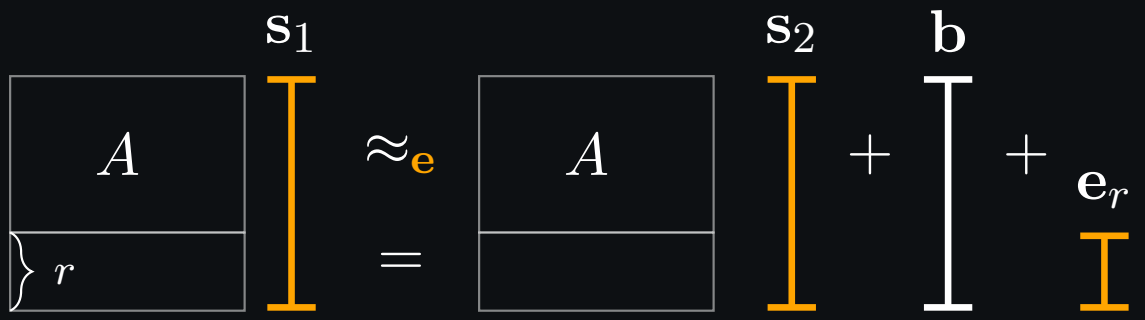


- Guess $r = \log_q R$ coordinates of $\mathbf{e}$, call it $\mathbf{e}_r$.
- $L_1 = \{\mathbf{s}_1 \in \{\pm 1, 0\}^n, \mathrm{Hash}(A\mathbf{s}_1)\}$
- $L_2 = \{\mathbf{s}_2 \in \{\pm 1, 0\}^n, \mathrm{Hash}(-A\mathbf{s}_2 + \mathbf{b})\}$
- Output $\mathbf{s}_1, \mathbf{s}_2$ s.t.
  - $\mathrm{Hash}(A\mathbf{s}_1) = \mathrm{Hash}(-A\mathbf{s}_2 + \mathbf{b})$ on $n - r$ coordinates
  - $A\mathbf{s}_1 = -A\mathbf{s}_2 + \mathbf{b} + \mathbf{e}_r$ on r coordinates

L_1 resp. L_2 are constructed in the meet-in-the-middle way by splitting $\mathbf{s}_1$ resp. $\mathbf{s}_2$ and using the equality condition on r.

$$\text{Time} \approx \text{Space} \approx (\underbrace{3^n}_{\text{search space}})^{1/4} \cdot \underbrace{3^r}_{\#\text{ guesses}}$$

Prior art II. May's idea



- Guess $r = \log_q R$ coordinates of $\mathbf{e}$, call it $\mathbf{e}_r$.
- $L_1 = \{\mathbf{s}_1 \in \{\pm 1, 0\}^n, \mathrm{Hash}(A\mathbf{s}_1)\}$
- $L_2 = \{\mathbf{s}_2 \in \{\pm 1, 0\}^n, \mathrm{Hash}(-A\mathbf{s}_2 + \mathbf{b})\}$
- Output $\mathbf{s}_1, \mathbf{s}_2$ s.t.
  - $\mathrm{Hash}(A\mathbf{s}_1) = \mathrm{Hash}(-A\mathbf{s}_2 + \mathbf{b})$ on $n - r$ coordinates
  - $A\mathbf{s}_1 = -A\mathbf{s}_2 + \mathbf{b} + \mathbf{e}_r$ on r coordinates

L_1 resp. L_2 are constructed in the meet-in-the-middle way by splitting $\mathbf{s}_1$ resp. $\mathbf{s}_2$ and using the equality condition on r.

Time $\approx$ Space $\approx$ ($\underbrace{3^n}_{\text{search space}}$)$^{1/4}$. $\cancel{\underbrace{3^r}_{\#\text{ guesses}}}$ $\leftarrow$ This work.

Our results

1. We improve May'21 combinatorial attack on ternary LWE achieving
$$\text{Time} \approx \text{Space} \approx (3^n)^{1/4}$$
2. We do so by combining the near neighbour search for ℓ_∞ with the representation technique
3. We present another approach that represents not only $\mathbf{s}$ but also $\mathbf{e}$ (see paper)
4. Runtime $(3^n)^{1/4}$ does not improve the asymptotics for interesting LWE parameters ($3^{\log_q R} = 2^{\mathcal{O}(n/\log n)}$) but makes a difference for exact bit complexities

Near Neighbour search in ℓ_∞

Close Pair problem in ℓ_∞: Given two equal-sized lists L_1, L_2 of iid. uniform random vectors in $\mathbb{Z}_q^n$ find almost all pairs $(\mathbf{x}_1, \mathbf{x}_2) \in L_1 \times L_2$, s.t.

$$|\mathbf{x}_1 - \mathbf{x}_2|_\infty = 1$$

Using Odlyzko's locality-sensitive hash function with parameter B

$$\begin{aligned} \mathrm{Hash}_{\mathbf{u}} : \mathbb{Z}_q^n &\to [0, \ldots, \left\lceil \frac{q}{B} \right\rceil - 1]^n \\ \mathbf{x} &\mapsto \left\lfloor \frac{x_i + u_i}{B} \right\rfloor_i \end{aligned}$$

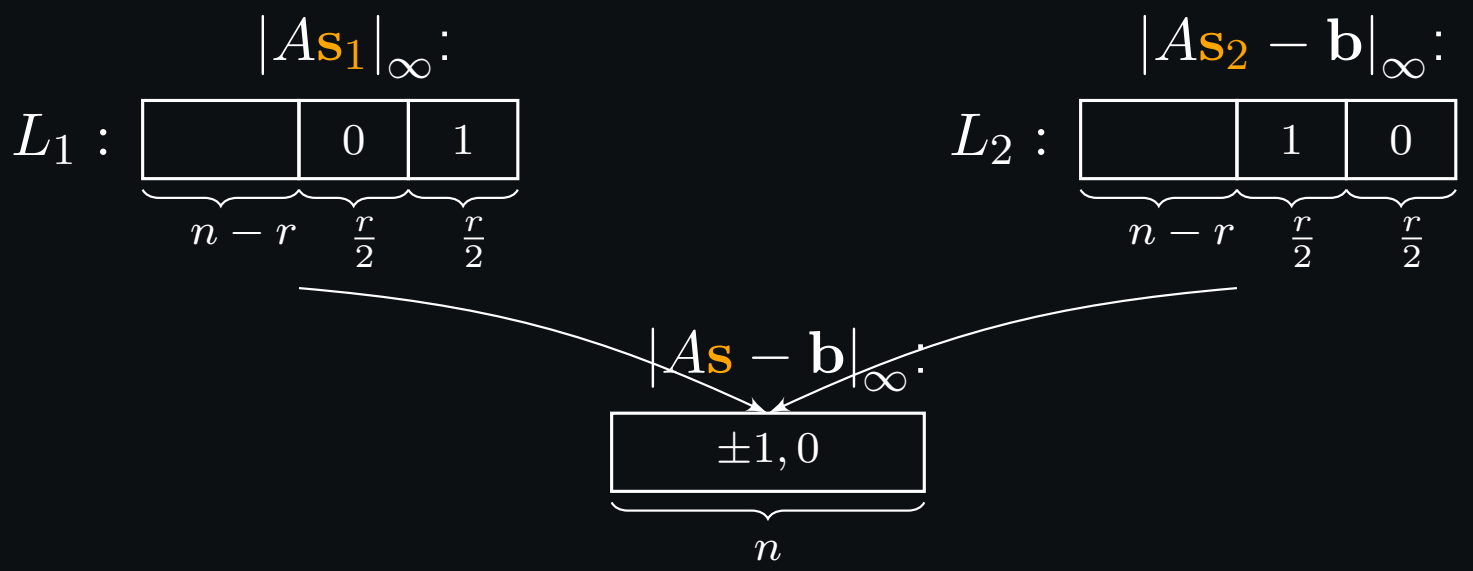
we can solve the close pair problem in

$$\mathrm{Time}_{\mathrm{LSH}} = |L_1|^2 \left(\frac{B^2}{(B-1)q} \right)^n \quad \mathrm{Space}_{\mathrm{LSH}} = |L_1|^2 \left(\frac{3}{q} \right)^n$$

The algorithm

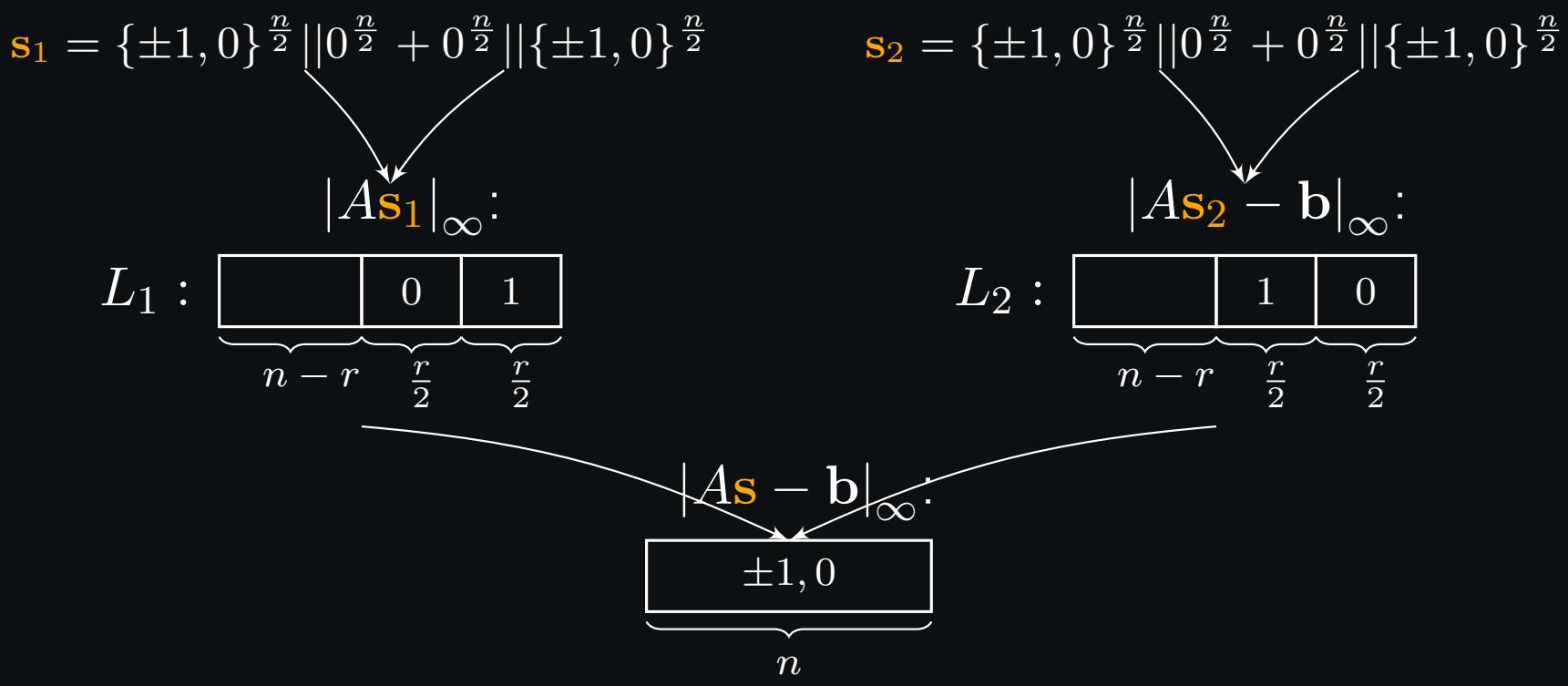
- Construct $L_1 = \{\mathbf{s}_1 \in \{\pm 1, 0\}^n : |A\mathbf{s}_1|_\infty$ has the "right pattern"$\}$
  $L_2 = \{\mathbf{s}_2 \in \{\pm 1, 0\}^n : |A\mathbf{s}_2|_\infty$ has the "right pattern"$\}$

The algorithm



- Construct $L_1 = \{\mathbf{s}_1 \in \{\pm 1, 0\}^n : |A\mathbf{s}_1|_\infty$ has the "right pattern"$\}$
  $L_2 = \{\mathbf{s}_2 \in \{\pm 1, 0\}^n : |A\mathbf{s}_2|_\infty$ has the "right pattern"$\}$
- Apply Near Neighbour search on $n - r$ first coordinates

The algorithm



- Use Near Neighbour search + Exact match to construct L_1, L_2 with the "right pattern"
- Construct $L_1 = \{\mathbf{s}_1 \in \{\pm 1, 0\}^n : |A\mathbf{s}_1|_\infty$ has the "right pattern"$\}$
  $L_2 = \{\mathbf{s}_2 \in \{\pm 1, 0\}^n : |A\mathbf{s}_2|_\infty$ has the "right pattern"$\}$
- Apply Near Neighbour search on $n - r$ first coordinates

Concrete results: bit securities

(n, q, ω)	New algorithm	Lattices[4]
NTRU Enc (NIST 3rd round candidate)		
$(509, 2048, 254)$	220	**124**
$(677, 2048, 254)$	256	**167**
$(821, 4096, 510)$	408	**197**
NTRU Prime (NIST 3rd round alternative)		
$(653, 4621, 288)$	262	**148**
$(761, 4591, 286)$	294	**174**
$(857, 5167, 322)$	337	**196**
NTRU IEEE-2008		
$(659, 2048, 76)$	**147**	151
$(761, 2048, 84)$	**159**	176
$(1087, 2048, 126)$	**236**	260
$(1499, 2048, 158)$	**311**	358

[4] D. Dachman-Soled, L. Ducas, H. Gong, M. Rossi. LWE with side information: Attacks and concrete security estimation. Crypto'20

Conclusion

- Combinatorial attacks on ternary LWE profit from sparse secretes. So avoid really sparse secrets in LWE/NTRU.
- Open question: is it possible to combine this combinatorial attack with lattice basis reduction (a.k.a. hybrid attack)?

`https://eprint.iacr.org/2021/1255`

Thank you!