

On the Quantitative Hardness of CVP

Huck Bennett,
Alexander Golovnev,
Noah Stephens-Davidowitz

NYCAC 2017

Outline

- Closest Vector Problem
- Applications
- Hardness
- Isolating Parallelepipeds

The Closest Vector Problem

Lattice

- A lattice \mathcal{L} is the set of all integer combinations of linearly independent basis vectors $\vec{b}_1, \dots, \vec{b}_n \in \mathbb{R}^d$

$$\mathcal{L} = \mathcal{L}(\vec{b}_1, \dots, \vec{b}_n) := \left\{ \sum_{i=1}^n z_i \vec{b}_i : z_i \in \mathbb{Z} \right\}$$

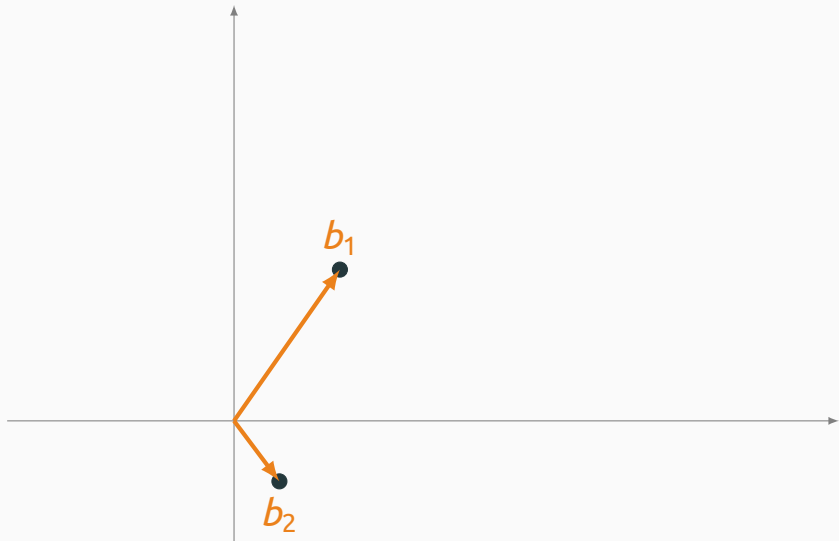
Lattice

- A lattice \mathcal{L} is the set of all integer combinations of linearly independent basis vectors $\vec{b}_1, \dots, \vec{b}_n \in \mathbb{R}^d$

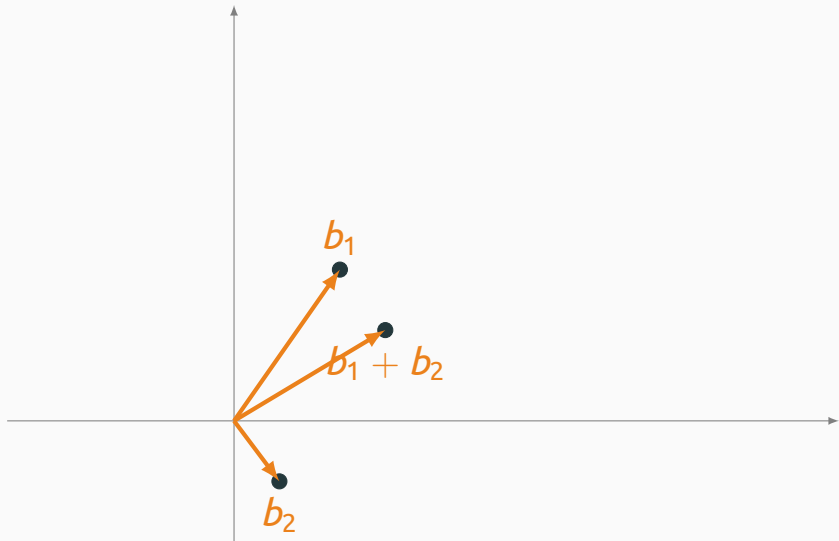
$$\mathcal{L} = \mathcal{L}(\vec{b}_1, \dots, \vec{b}_n) := \left\{ \sum_{i=1}^n z_i \vec{b}_i : z_i \in \mathbb{Z} \right\}$$

- n is the **rank** of \mathcal{L} , d is the (ambient) **dimension**

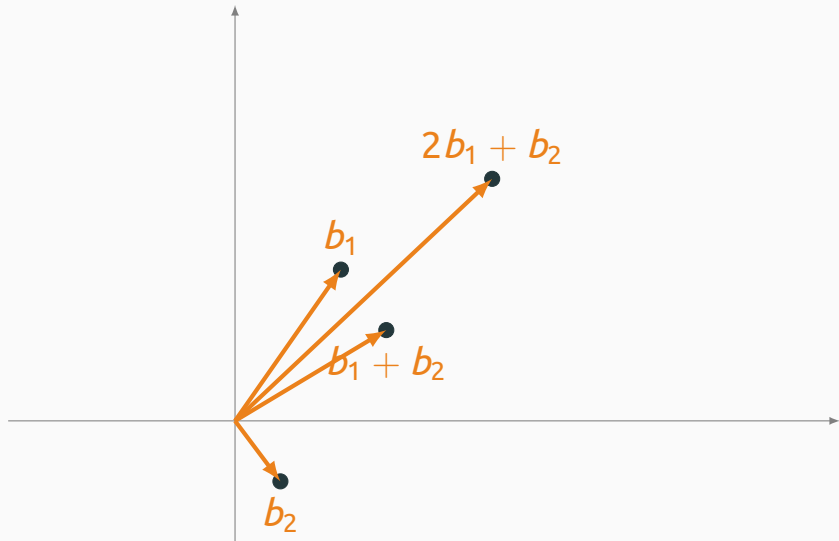
Lattice. Example



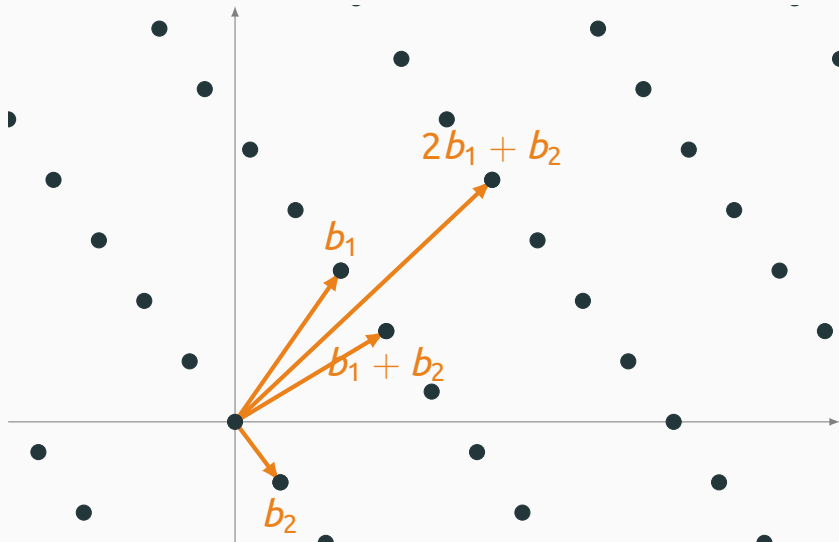
Lattice. Example



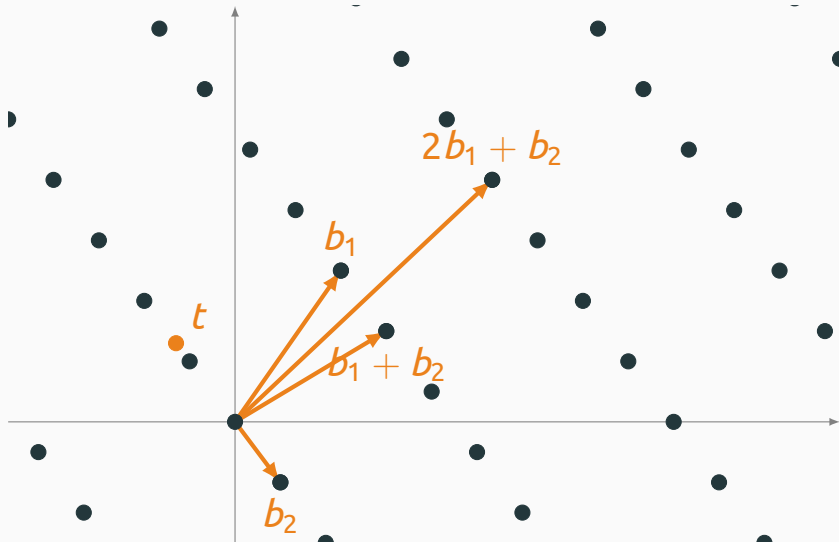
Lattice. Example



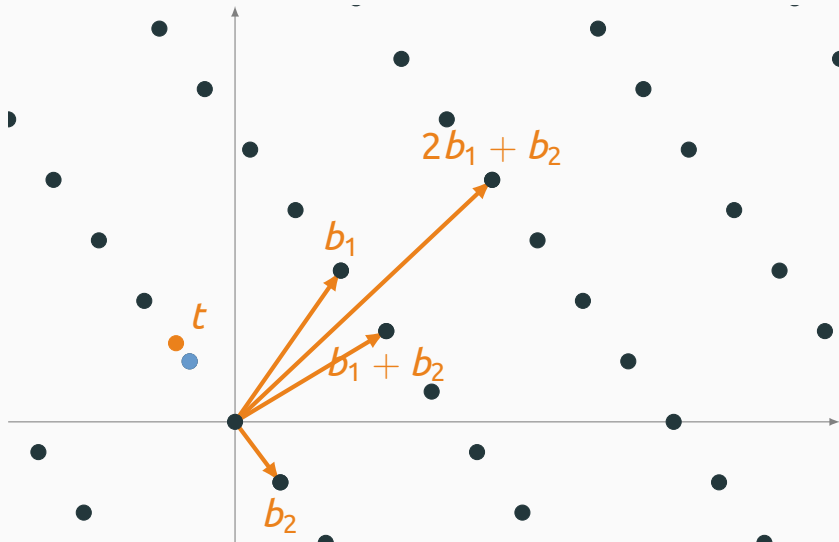
Lattice. Example



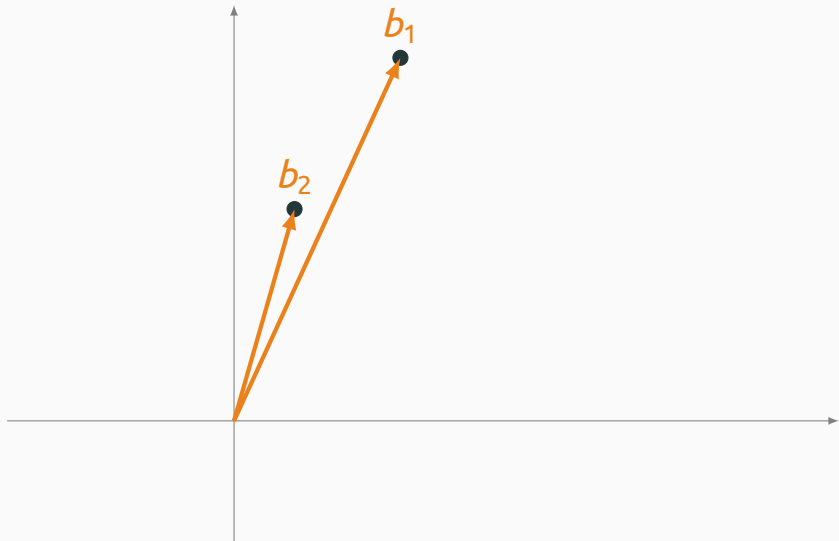
Lattice. Example



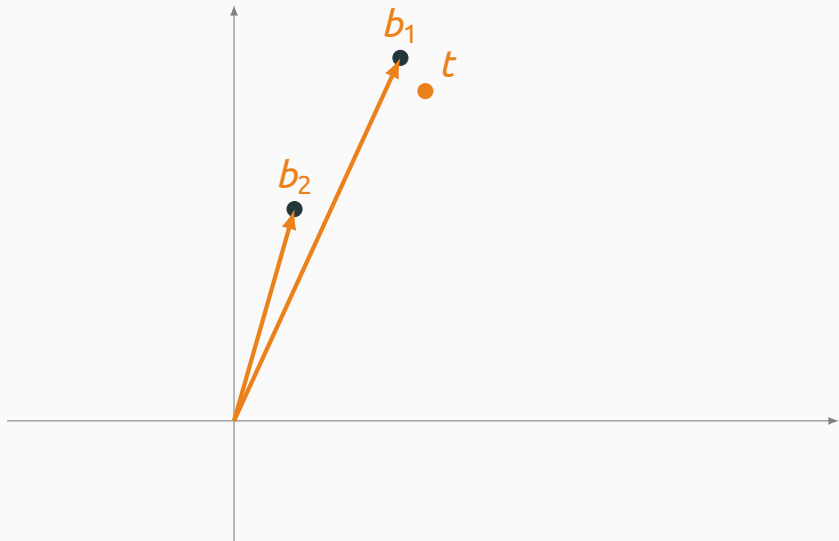
Lattice. Example



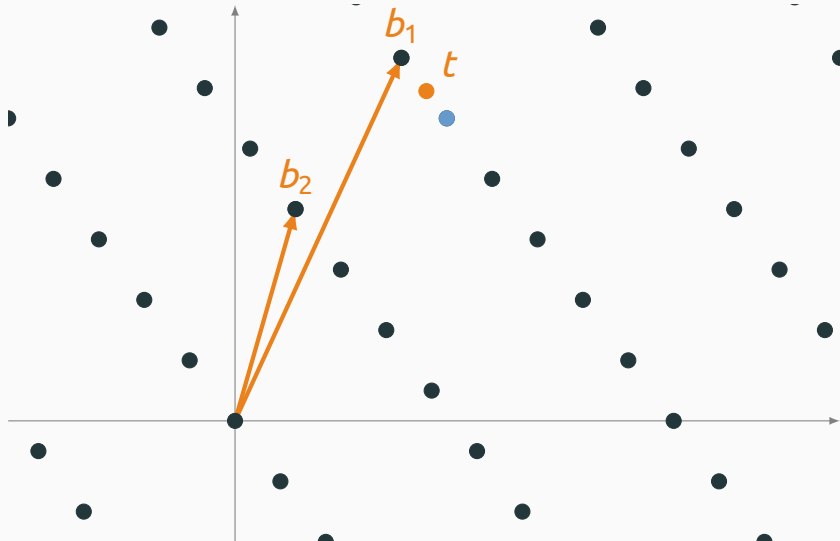
Lattice. Example



Lattice. Example



Lattice. Example



Closest Vector Problem

- Given a **basis** for a $\mathcal{L} \subset \mathbb{R}^d$ and a **target** $t \in \mathbb{R}^d$, compute the **distance** from t to \mathcal{L}

Closest Vector Problem

- Given a **basis** for a $\mathcal{L} \subset \mathbb{R}^d$ and a **target** $t \in \mathbb{R}^d$, compute the **distance** from t to \mathcal{L}
- Distance is defined in terms of the ℓ_p norm; for $1 \leq p < \infty$:

$$\|\vec{x}\|_p := (|x_1|^p + |x_2|^p + \cdots + |x_d|^p)^{1/p}$$

for $p = \infty$:

$$\|\vec{x}\|_\infty := \max_{1 \leq i \leq d} |x_i|$$

Closest Vector Problem

- Given a **basis** for a $\mathcal{L} \subset \mathbb{R}^d$ and a **target** $t \in \mathbb{R}^d$, compute the **distance** from t to \mathcal{L}
- Distance is defined in terms of the ℓ_p norm; for $1 \leq p < \infty$:

$$\|\vec{x}\|_p := (|x_1|^p + |x_2|^p + \cdots + |x_d|^p)^{1/p}$$

for $p = \infty$:

$$\|\vec{x}\|_\infty := \max_{1 \leq i \leq d} |x_i|$$

- **CVP_p**—Closest Vector Problem in the ℓ_p norm

Applications

Applications

- Factoring polynomials over the rationals
[LLL'82]

Applications

- Factoring polynomials over the rationals
[LLL'82]
- Integer Programming [Len83,Kan87,DPV11]

Applications

- Factoring polynomials over the rationals
[LLL'82]
- Integer Programming [Len83,Kan87,DPV11]
- Cryptanalysis [Odl90,JS98,NS01]

Lattice-Based Cryptography

- Conjectured Quantum Security

Lattice-Based Cryptography

- Conjectured Quantum Security
- Efficiency, Parallelism, Simplicity

Lattice-Based Cryptography

- Conjectured Quantum Security
- Efficiency, Parallelism, Simplicity
- Worst-Case Hardness Proofs

Lattice-Based Cryptography

- Conjectured Quantum Security
- Efficiency, Parallelism, Simplicity
- Worst-Case Hardness Proofs
- Powerful Cryptography: FHE, ABE

Lattice-Based Cryptography

- Conjectured Quantum Security
- Efficiency, Parallelism, Simplicity
- Worst-Case Hardness Proofs
- Powerful Cryptography: FHE, ABE
- About to be Deployed

Real Life Cryptography

GitHub, Inc. [US] <https://github.com/lwe-frodo>



This organization

Search

Pull requests

Issues

Marketplace

Gist



lwe-frodo ⓘ

Post-quantum key exchange from the learning with errors problem

<https://eprint.iacr.org/2016/659>

📁 **Repositories**

👤 **People** 0

Search repositories...

lwe-frodo

Post-quantum key exchange from the learning with errors problem — from the paper "Frodo: Take off the ring! Practical, Quantum-Secure Key Exchange from LWE", published in ACM CCS 2016, <https://eprint.iacr.org/2016/659>

🔒 cryptography

🔒 post-quantum-cryptography

🔒 key-exchange-algorithms

🌐 C ★ 19 🍷 4 Updated on Oct 17, 2016

Real Life Cryptography

GitHub, Inc. [US] <https://github.com/lwe-frodo>



This organization

Search

NEWS -- December 15, 2016: The National Institute of Standards and Technology (NIST) is now accepting submissions for quantum-resistant public-key cryptographic algorithms. The deadline for submission is **November 30, 2017**. Please see the Post-Quantum Cryptography Standardization menu at left for the complete submission requirements and evaluation criteria.

Repositories

People 0

Search repositories...

lwe-frodo

Post-quantum key exchange from the learning with errors problem — from the paper "Frodo: Take off the ring! Practical, Quantum-Secure Key Exchange from LWE", published in ACM CCS 2016, <https://eprint.iacr.org/2016/659>

cryptography

post-quantum-cryptography

key-exchange-algorithms

● ★ 19 🍴 4 Updated on Oct 17, 2016

Real Life Cryptography

GitHub, Inc. [US] <https://github.com/lwe-frodo>



This organization Search

NEWS -- December 15, 2016: The National Institute of Standards and Technology (NIST) is now accepting submissions for quantum-resistant public-key cryptographic algorithms. The deadline for submission is **November 30, 2017**. Please see the Post-Quantum Cryptography Standardization menu at left for complete submission requirements and evaluation criteria.

WIRED

repositories

People 0

Google Tests New Crypto in Chrome to Fend Off Quantum Attacks

GOOGLE TESTS NEW CRYPTO IN CHROME TO FEND OFF QUANTUM ATTACKS

Post-quantum key exchange from the paper "Frodo: Take off the ring! Practical, Secure Key Exchange from LWE", published in ACM CCS 2016, <https://eprint.iacr.org/2016/659>

cryptography

post-quantum-cryptography

key-exchange-algorithms

● C ★ 19 🍷 4 Updated on Oct 17, 2016

Hardness

Hardness of CVP

- CVP_ρ is NP-hard for every $1 \leq \rho \leq \infty$ [vEB81]

Hardness of CVP

- CVP_ρ is NP-hard for every $1 \leq \rho \leq \infty$ [vEB81]
- CVP_2 can be solved in $2^{n+o(n)}$ time [ADS15]

Hardness of CVP

- CVP_ρ is NP-hard for every $1 \leq \rho \leq \infty$ [vEB81]
- CVP_2 can be solved in $2^{n+o(n)}$ time [ADS15]
- Cryptographic applications require **quantitative** hardness of CVP [ADPS16,BCD+16,NIS16]:
a $2^{n/20}$ -time algorithm would break these schemes in practice

k-SAT

$$\blacksquare (x_1 \vee \neg x_2 \vee \dots \vee x_k) \wedge \dots \wedge (x_7 \vee \neg x_4 \vee \dots \vee x_3)$$

k -SAT

- $(x_1 \vee \neg x_2 \vee \dots \vee x_k) \wedge \dots \wedge (x_7 \vee \neg x_4 \vee \dots \vee x_3)$
- n Boolean vars, m clauses, clause length $\leq k$

k -SAT

- $(x_1 \vee \neg x_2 \vee \dots \vee x_k) \wedge \dots \wedge (x_7 \vee \neg x_4 \vee \dots \vee x_3)$
- n Boolean vars, m clauses, clause length $\leq k$
- SETH [IP99]. There exists a constant k :
no algorithm solves k -SAT in $2^{0.99n}$ time

k -SAT

- $(x_1 \vee \neg x_2 \vee \dots \vee x_k) \wedge \dots \wedge (x_7 \vee \neg x_4 \vee \dots \vee x_3)$
- n Boolean vars, m clauses, clause length $\leq k$
- SETH [IP99]. There exists a constant k :
no algorithm solves k -SAT in $2^{0.99n}$ time
- Goal: Reduce k -SAT on n vars
to CVP on a rank- n lattice

A Very Special Case: 2-SAT

	x_1	x_2	\dots	x_{n-1}	x_n	
x_1	2α	0	\dots	0	0	α
x_2	0	2α	\dots	0	0	α
\vdots	\vdots	\vdots	\ddots	0	0	\vdots
x_n	0	0	\dots	0	2α	α
$C_1 = (x_1 \vee x_2)$	2	2	\dots	0	0	3
$C_2 = (x_1 \vee x_n)$	2	0	\dots	0	2	3
\vdots	\vdots	\vdots	\ddots	\vdots	\vdots	\vdots
$C_m = (x_{n-1} \vee x_n)$	0	0	\dots	2	2	3

A Very Special Case: 2-SAT

	x_1	x_2	\dots	x_{n-1}	x_n	
x_1	2α	0	\dots	0	0	α
x_2	0	2α	\dots	0	0	α
\vdots	\vdots	\vdots	\ddots	0	0	\vdots
x_n	0	0	\dots	0	2α	α
$C_1 = (x_1 \vee x_2)$	2	2	\dots	0	0	3
$C_2 = (x_1 \vee x_n)$	2	0	\dots	0	2	3
\vdots	\vdots	\vdots	\ddots	\vdots	\vdots	\vdots
$C_m = (x_{n-1} \vee x_n)$	0	0	\dots	2	2	3

A Very Special Case: 2-SAT

	x_1	x_2	\dots	x_{n-1}	x_n	
x_1	2α	0	\dots	0	0	α
x_2	0	2α	\dots	0	0	α
\vdots	\vdots	\vdots	\ddots	0	0	\vdots
x_n	0	0	\dots	0	2α	α
$C_1 = (x_1 \vee x_2)$	2	2	\dots	0	0	3
$C_2 = (x_1 \vee x_n)$	2	0	\dots	0	2	3
\vdots	\vdots	\vdots	\ddots	\vdots	\vdots	\vdots
$C_m = (x_{n-1} \vee x_n)$	0	0	\dots	2	2	3

A Very Special Case: 2-SAT. Proof

x_1	x_2	\dots	x_{n-1}	x_n
2α	0	\dots	0	0
0	2α	\dots	0	0
\vdots	\vdots	\ddots	0	0
0	0	\dots	0	2α
2	2	\dots	0	0
2	0	\dots	0	2
\vdots	\vdots	\ddots	\vdots	\vdots
0	0	\dots	2	2

α
α
\vdots
α
3
3
\vdots
3

α is very large

A Very Special Case: 2-SAT. Proof

x_1	x_2	\dots	x_{n-1}	x_n	
2α	0	\dots	0	0	α
0	2α	\dots	0	0	α
\vdots	\vdots	\ddots	0	0	\vdots
0	0	\dots	0	2α	α
2	2	\dots	0	0	3
2	0	\dots	0	2	3
\vdots	\vdots	\ddots	\vdots	\vdots	\vdots
0	0	\dots	2	2	3

α is very large
If $x \in \{0, 1\}^n$,
first n lines give
distance $n\alpha^p$

A Very Special Case: 2-SAT. Proof

x_1	x_2	\dots	x_{n-1}	x_n
2α	0	\dots	0	0
0	2α	\dots	0	0
\vdots	\vdots	\ddots	0	0
0	0	\dots	0	2α
2	2	\dots	0	0
2	0	\dots	0	2
\vdots	\vdots	\ddots	\vdots	\vdots
0	0	\dots	2	2

α
 α
 \vdots
 α

α is very large
 If $x \in \{0, 1\}^n$,
 first n lines give
 distance $n\alpha^p$
 If $x \notin \{0, 1\}^n$,
 distance is
 $\geq (n+1)\alpha^p$

A Very Special Case: 2-SAT. Proof

x_1	x_2	\dots	x_{n-1}	x_n		
2α	0	\dots	0	0	α	$x \in \{0, 1\}^n$
0	2α	\dots	0	0	α	
\vdots	\vdots	\ddots	0	0	\vdots	
0	0	\dots	0	2α	α	
2	2	\dots	0	0	3	
2	0	\dots	0	2	3	
\vdots	\vdots	\ddots	\vdots	\vdots	\vdots	
0	0	\dots	2	2	3	

A Very Special Case: 2-SAT. Proof

x_1	x_2	\dots	x_{n-1}	x_n	
2α	0	\dots	0	0	α
0	2α	\dots	0	0	α
\vdots	\vdots	\ddots	0	0	\vdots
0	0	\dots	0	2α	α
2	2	\dots	0	0	3
2	0	\dots	0	2	3
\vdots	\vdots	\ddots	\vdots	\vdots	\vdots
0	0	\dots	2	2	3

$x \in \{0, 1\}^n$
sat clause contributes 1 to the distance

A Very Special Case: 2-SAT. Proof

x_1	x_2	\dots	x_{n-1}	x_n	
2α	0	\dots	0	0	α $x \in \{0, 1\}^n$
0	2α	\dots	0	0	α
\vdots	\vdots	\ddots	0	0	\vdots
0	0	\dots	0	2α	α
2	2	\dots	0	0	3
2	0	\dots	0	2	3
\vdots	\vdots	\ddots	\vdots	\vdots	\vdots
0	0	\dots	2	2	3

sat clause contributes 1 to the distance
 unsat clause contributes $3^p > 1$

MAX-2-SAT

- Given an instance of 2-SAT, we construct an instance of CVP_ρ , s.t.
 - If all clauses are **sat** —distance is **small**
 - If **not** all clauses are **sat** —distance is **large**

MAX-2-SAT

- Given an instance of 2-SAT, we construct an instance of CVP_ρ , s.t.
 - If all clauses are **sat** —distance is **small**
 - If **not** all clauses are **sat** —distance is **large**
- Actually, the reduction gives **the number of satisfiable clauses**

MAX-2-SAT

- Given an instance of 2-SAT, we construct an instance of CVP_ρ , s.t.
 - If all clauses are **sat** —distance is **small**
 - If **not** all clauses are **sat** —distance is **large**
- Actually, the reduction gives **the number of satisfiable clauses**
- This is an NP-hard problem MAX-2-SAT

MAX-2-SAT

- Given an instance of 2-SAT, we construct an instance of CVP_ρ , s.t.
 - If all clauses are **sat** —distance is **small**
 - If **not** all clauses are **sat** —distance is **large**
- Actually, the reduction gives **the number of satisfiable clauses**
- This is an NP-hard problem MAX-2-SAT
- Best algorithm for MAX-2-SAT runs in $2^{\omega n/3} < 1.74^n$

Generalization to k -SAT?

- For all values of k , we want to reduce k -SAT to CVP_ρ

Generalization to k -SAT?

- For all values of k , we want to reduce k -SAT to CVP_ρ
- This would give 1.99^n -hardness of CVP_ρ under SETH

Generalization to k -SAT?

- For **all values of k** , we want to reduce k -SAT to CVP_ρ
- This would give **1.99^n** -hardness of CVP_ρ under SETH
- A 2-SAT clause is sat iff # of sat literals is 1 or 2

Generalization to k -SAT?

- For **all values of k** , we want to reduce k -SAT to CVP_ρ
- This would give **1.99^n** -hardness of CVP_ρ under SETH
- A 2-SAT clause is sat iff # of sat literals is 1 or 2
- 2 and 4 are equidistant from 3!

Generalization to k -SAT?

- For **all values of k** , we want to reduce k -SAT to CVP_ρ
- This would give **1.99^n** -hardness of CVP_ρ under SETH
- A 2-SAT clause is sat iff # of sat literals is 1 or 2
- 2 and 4 are equidistant from 3!
- For k -SAT, we can't find k numbers which are equidistant from some other number...

Generalization to k -SAT!

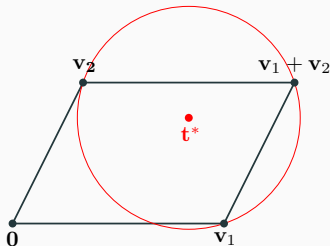
- We can find k **vectors** which are equidistant from some other vector!

Generalization to k -SAT!

- We can find k **vectors** which are equidistant from some other vector!
- **Goal:** Find k vectors $V = (\vec{v}_1, \dots, \vec{v}_k) \in \mathbb{R}^{m \times k}$ and $\vec{t} \in \mathbb{R}^m$, s.t.
 - for all non-zero $\vec{y} \in \{0, 1\}^k$, $\|V\vec{y} - \vec{t}\|_p = 1$
 - for $\vec{y} = 0^k$, $\|V\vec{y} - \vec{t}\|_p = \|\vec{t}\|_p > 1$

Generalization to k -SAT!

- We can find k **vectors** which are equidistant from some other vector!
- **Goal:** Find k vectors $V = (\vec{v}_1, \dots, \vec{v}_k) \in \mathbb{R}^{m \times k}$ and $\vec{t} \in \mathbb{R}^m$, s.t.
 - for all non-zero $\vec{y} \in \{0, 1\}^k$, $\|V\vec{y} - \vec{t}\|_p = 1$
 - for $\vec{y} = 0^k$, $\|V\vec{y} - \vec{t}\|_p = \|\vec{t}\|_p > 1$



Isolating Parallelepipeds

Isolating Parallelepipeds

Definition (Isolating Parallelepipeds)

k vectors $V = (\vec{v}_1, \dots, \vec{v}_k) \in \mathbb{R}^{m \times k}$ and $\vec{t} \in \mathbb{R}^m$

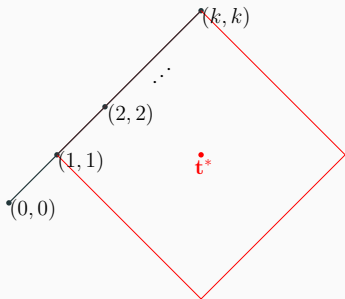
- for all non-zero $\vec{y} \in \{0, 1\}^k$, $\|V\vec{y} - \vec{t}\|_\rho = 1$
- for $\vec{y} = 0^k$, $\|V\vec{y} - \vec{t}\|_\rho = \|\vec{t}\|_\rho > 1$

Isolating Parallelepipeds in ℓ_1

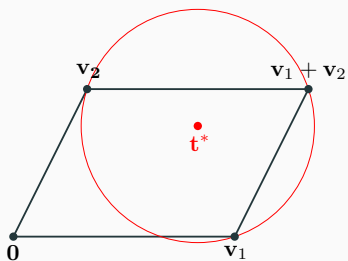
Definition (Isolating Parallelepipeds)

k vectors $V = (\vec{v}_1, \dots, \vec{v}_k) \in \mathbb{R}^{m \times k}$ and $\vec{t} \in \mathbb{R}^m$

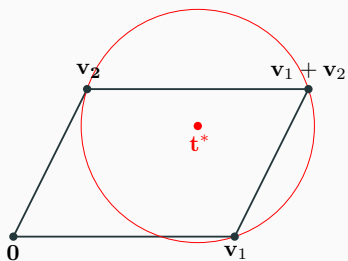
- for all non-zero $\vec{y} \in \{0, 1\}^k$, $\|V\vec{y} - \vec{t}\|_p = 1$
- for $\vec{y} = 0^k$, $\|V\vec{y} - \vec{t}\|_p = \|\vec{t}\|_p > 1$



Isolating Parallelepipeds in ℓ_2

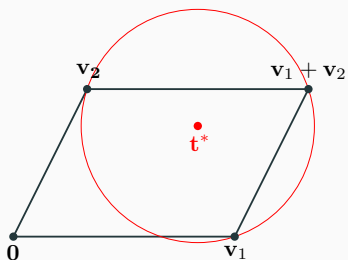


Isolating Parallelepipeds in ℓ_2



Can we do for 3 vectors?

Isolating Parallelepipeds in ℓ_2



Can we do for 3 vectors? **No!**

Isolating Parallelepipeds

- If ρ is an odd integer, then IPs always exist

Isolating Parallelepipeds

- If ρ is an odd integer, then IPs always exist
- If ρ is an even integer, then IPs exist only for at most $k \leq \rho$ vectors

Isolating Parallelepipeds

- If ρ is an odd integer, then IPs always exist
- If ρ is an even integer, then IPs exist only for at most $k \leq \rho$ vectors
- For any k and any $\rho = \rho_0 + \delta(n)$ with $\delta(n) \neq 0$ and $\delta(n) \rightarrow 0$, they exist for sufficiently large n

Isolating Parallelepipeds

- If p is an odd integer, then IPs always exist
- If p is an even integer, then IPs exist only for at most $k \leq p$ vectors
- For any k and any $p = p_0 + \delta(n)$ with $\delta(n) \neq 0$ and $\delta(n) \rightarrow 0$, they exist for sufficiently large n
- For any fixed k , IPs exist for all but finitely many values of p

Candidate for odd ρ

$$V := \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -1 & 1 \\ -1 & 1 & 1 \\ 1 & -1 & -1 \\ -1 & 1 & -1 \\ -1 & -1 & 1 \\ -1 & -1 & -1 \end{pmatrix}, \quad \vec{t} := \begin{pmatrix} t \\ t \\ t \\ t \\ t \\ t \\ t \\ t \end{pmatrix}.$$

Candidate for odd ρ

$$V := \begin{matrix} \alpha_3 \times \\ \alpha_2 \times \\ \alpha_2 \times \\ \alpha_2 \times \\ \alpha_1 \times \\ \alpha_1 \times \\ \alpha_1 \times \\ \alpha_0 \times \end{matrix} \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -1 & 1 \\ -1 & 1 & 1 \\ 1 & -1 & -1 \\ -1 & 1 & -1 \\ -1 & -1 & 1 \\ -1 & -1 & -1 \end{pmatrix}, \quad \vec{t} := \begin{pmatrix} t \\ t \\ t \\ t \\ t \\ t \\ t \\ t \end{pmatrix}.$$

Constraints for odd ρ

- This gives a system of k linear equations on $\alpha_1, \dots, \alpha_k$

Constraints for odd ρ

- This gives a system of k linear equations on $\alpha_1, \dots, \alpha_k$
- But we need a solution with all α 's **non-negative**

Constraints for odd ρ

- This gives a system of k linear equations on $\alpha_1, \dots, \alpha_k$
- But we need a solution with all α 's **non-negative**
- $M \in \mathbb{R}(t)^{k \times k}, \alpha = (\alpha_1, \dots, \alpha_k) \in \mathbb{R}^k :$

$$M \cdot \alpha = \begin{pmatrix} 1 + \varepsilon \\ 1 \\ \vdots \\ 1 \end{pmatrix}$$

Odd ρ . Proof

- M is stochastic with a positive eigenvalue, so it suffices to show M is invertible:

Odd ρ . Proof

- M is stochastic with a positive eigenvalue, so it suffices to show M is invertible:
 - Let $\alpha' = M^{-1} \cdot e_1$

Odd ρ . Proof

- M is stochastic with a positive eigenvalue, so it suffices to show M is invertible:
 - Let $\alpha' = M^{-1} \cdot e_1$
 - $\alpha = \delta_1 \cdot \alpha' + \delta_2 \cdot \mathbf{1}_k$

Odd ρ . Proof

- M is stochastic with a positive eigenvalue, so it suffices to show M is invertible:
 - Let $\alpha' = M^{-1} \cdot e_1$
 - $\alpha = \delta_1 \cdot \alpha' + \delta_2 \cdot \mathbf{1}_k$
 - $M \cdot \alpha = (1 + \varepsilon, 1, \dots, 1)^T$

Odd ρ . Proof

- M is stochastic with a positive eigenvalue, so it suffices to show M is invertible:
 - Let $\alpha' = M^{-1} \cdot e_1$
 - $\alpha = \delta_1 \cdot \alpha' + \delta_2 \cdot \mathbf{1}_k$
 - $M \cdot \alpha = (1 + \varepsilon, 1, \dots, 1)^T$
- $\det(M)$ is a piecewise combination of polynomials of degree $(k + 1)\rho$

Odd ρ . Proof

- M is stochastic with a positive eigenvalue, so it suffices to show M is invertible:
 - Let $\alpha' = M^{-1} \cdot e_1$
 - $\alpha = \delta_1 \cdot \alpha' + \delta_2 \cdot \mathbf{1}_k$
 - $M \cdot \alpha = (1 + \varepsilon, 1, \dots, 1)^T$
- $\det(M)$ is a piecewise combination of polynomials of degree $(k + 1)\rho$
- We show that at least one of these polynomials is non-zero

Conclusions

- Isolating Parallelepipeds don't exist for even ρ , and exist for almost any other ρ
 - If SETH holds, no $2^{0.99n}$ -algorithm solves CVP_ρ for these values of ρ

Conclusions

- Isolating Parallelepipeds don't exist for even ρ , and exist for almost any other ρ
 - If SETH holds, no $2^{0.99n}$ -algorithm solves CVP_ρ for these values of ρ
- Other hardness results for lattice problems
 - $\text{SVP}_\infty, \text{CVPP}_\rho, \dots$

Conclusions

- Isolating Parallelepipeds don't exist for even ρ , and exist for almost any other ρ
 - If SETH holds, no $2^{0.99n}$ -algorithm solves CVP_ρ for these values of ρ
- Other hardness results for lattice problems
 - $\text{SVP}_\infty, \text{CVPP}_\rho, \dots$
- Even hardness of approximation under Gap-ETH for all ρ

Thank you for your attention!