Progressive lattice sieving

Thijs Laarhoven and Artur Mariano

mail@thijs.com
http://www.thijs.com/

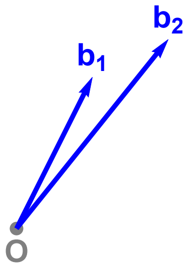
PQCrypto 2018, Fort Lauderdale (FL), USA
(April 10, 2018)

Lattices

What is a lattice?

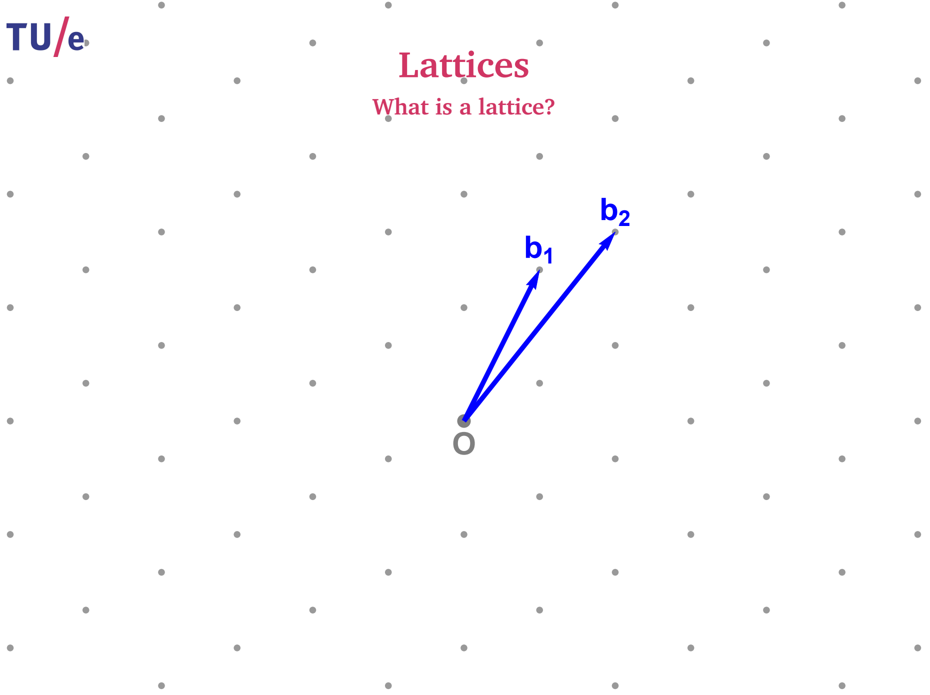
Lattices

What is a lattice?



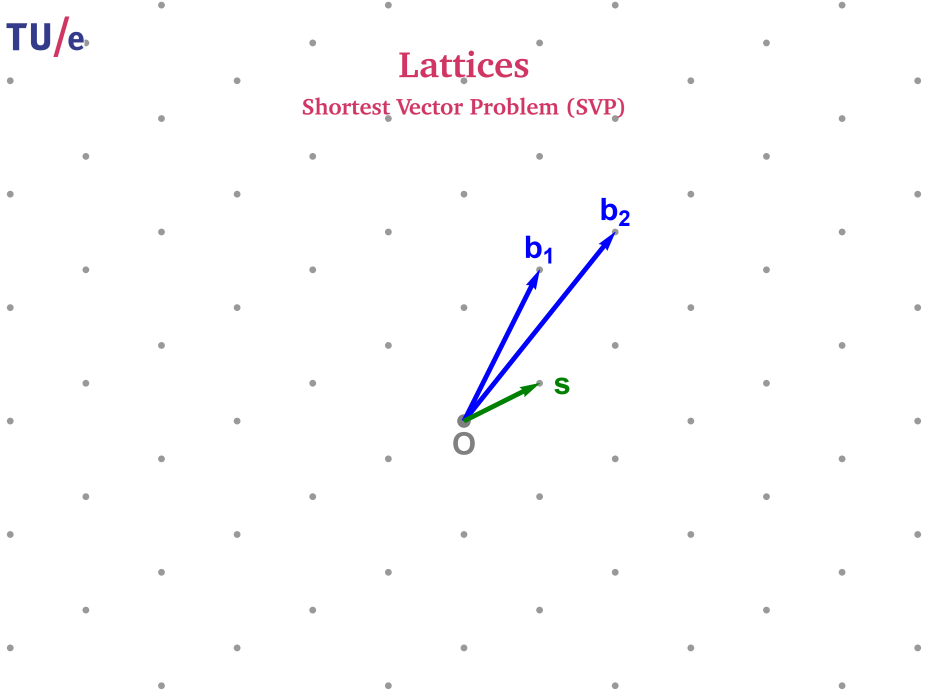
Lattices

What is a lattice?



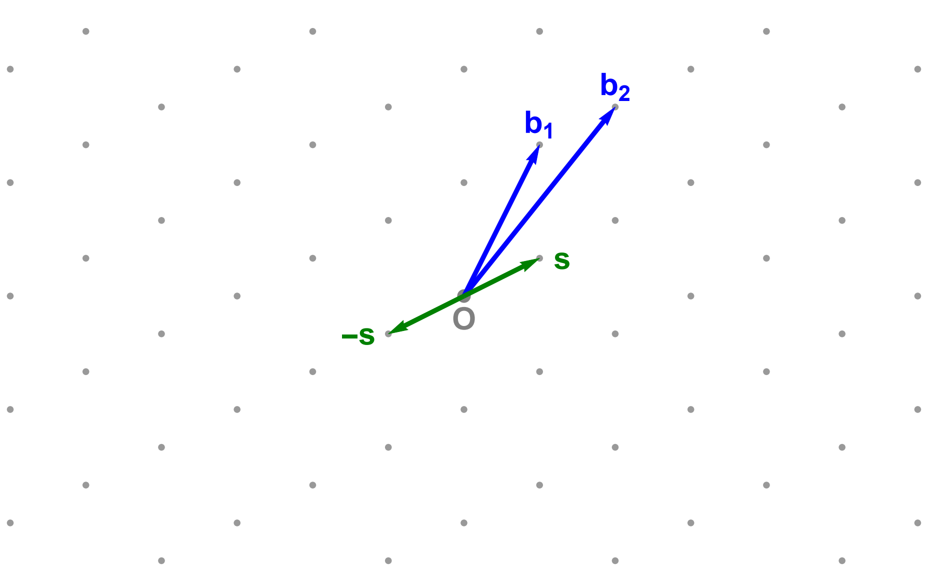
Lattices

Shortest Vector Problem (SVP)



Lattices

Shortest Vector Problem (SVP)



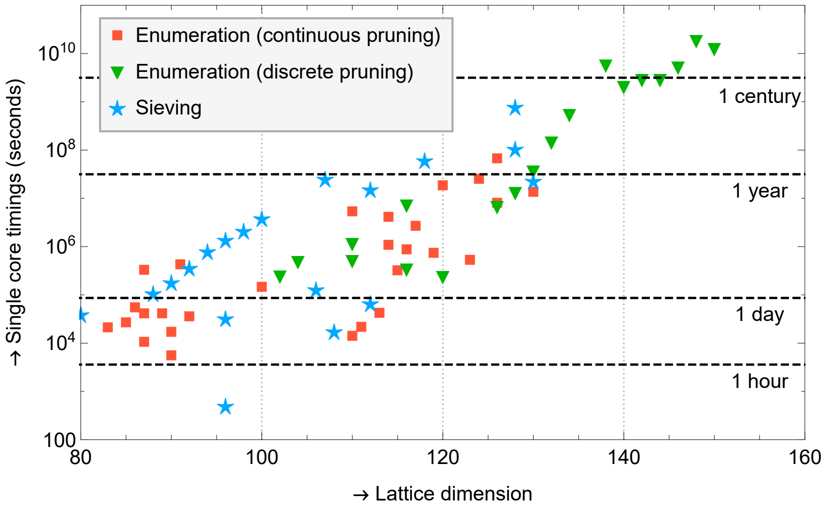
SVP hardness

Theory

	Algorithm	$\log_2$(Time)	$\log_2$(Space)
Proven SVP	Enumeration [Poh81, Kan83, ..., MW15, AN17]	$O(n \log n)$	$O(\log n)$
	AKS-sieve [AKS01, NV08, MV10, HPS11]	$3.398n$	$1.985n$
	ListSieve [MV10, MDB14]	$3.199n$	$1.327n$
	Birthday sieves [PS09, HPS11]	$2.465n$	$1.233n$
	Enumeration/DGS hybrid [CCL17]	$2.048n$	$0.500n$
	Voronoi cell algorithm [AEVZ02, MV10b]	$2.000n$	$1.000n$
	Quantum sieve [LMP13, LMP15]	$1.799n$	$1.286n$
	Quantum enum/DGS [CCL17]	$1.256n$	$\mathbf{0.500n}$
	Discrete Gaussian sampling [ADRS15, ADS15, AS18]	$\mathbf{1.000n}$	$1.000n$
Heuristic SVP	The Nguyen–Vidick sieve [NV08]	$0.415n$	$0.208n$
	The GaussSieve [MV10, ..., IKMT14, BNvdP16, YKYC17]	$0.415n$	$0.208n$
	Triple sieve [BLS16, HK17]	$0.396n$	$0.189n$
	Two-level sieve [WLTB11]	$0.384n$	$0.256n$
	Three-level sieve [ZPH13]	$0.3778n$	$0.283n$
	Overlattice sieve [BGJ14]	$0.3774n$	$0.293n$
	Triple sieve with NNS [HK17, HKL18]	$0.359n$	$\mathbf{0.189n}$
	Hyperplane LSH [Cha02, Laa15, ..., LM18, Duc18]	$0.337n$	$0.337n$
	Graph-based NNS [EPY99, DCL11, MPLK14, Laa18]	$0.327n$	$0.282n$
	Hypercube LSH [TT07, Laa17]	$0.322n$	$0.322n$
	Quantum sieve [LMP13, LMP15]	$0.312n$	$0.208n$
	May–Ozerov NNS [MO15, BGJ15]	$0.311n$	$0.311n$
	Spherical LSH [AINR14, LdW15]	$0.298n$	$0.298n$
	Cross-polytope LSH [TT07, AILRS15, BL16, KW17]	$0.298n$	$0.298n$
	Spherical LSF [BDGL16, MLB17, ALRW17, Chr17]	$\mathbf{0.292n}$	$0.292n$
	Quantum NNS sieve [LMP15, Laa16]	$\mathbf{0.265n}$	$0.265n$

SVP hardness

Practice [SVP17]



SVP hardness

NIST submissions

Title	Si	En	Submitters
CRYSTALS–Dilithium	•		**Lyubashevsky**, Ducas, Kiltz, Lepoint, Schwabe, Seiler, Stehlé
CRYSTALS–Kyber	•		**Schwabe**, Avanzi, Bos, Ducas, Kiltz, Lepoint, Lyubashevsky, Schanck, ...
Ding Key Exchange	•		**Ding**, Takagi, Gao, Wang
(R.)EMBLEM	•		**Seo**, Park, Lee, Kim, Lee
FALCON	•		**Prest**, Fouque, Hoffstein, Kirchner, Lyubashevsky, Pornin, Ricosset, ...
FrodoKEM	•		**Naehrig**, Alkim, Bos, Ducas, Easterbrook, LaMacchia, Longa, Mironov, ...
Giophantus	•		**Akiyama**, Goto, Okumura, Takagi, Nuida, Hanaoka, Shimizu, Ikematsu
HILA5	•		**Saarinen**
KCL	•		**Zhao**, Jin, Gong, Sui
KINDI	•		**El Bansarkhani**
LAC	•		**Lu**, Liu, Jia, Xue, He, Zhang
LIMA	•		**Smart**, Albrecht, Lindell, Orsini, Osheter, Paterson, Peer
Lizard	•		**Cheon**, Park, Lee, Kim, Song, Hong, Kim, Kim, Hong, Yun, Kim, Park, ...
LOTUS		•	**Phong**, Hayashi, Aono, Moriai
NewHope	•		**Pöppelmann**, Alkim, Avanzi, Bos, Ducas, De La Piedra, Schwabe, Stebila
NTRUEncrypt	◦	◦	**Zhang**, Chen, Hoffstein, Whyte
NTRU-HRSS-KEM	•		**Schanck**, Hülsing, Rijneveld, Schwabe
NTRU Prime		•	**Bernstein**, Chuengsatiansup, Lange, Van Vredendaal
pqNTRUSign	◦	◦	**Zhang**, Chen, Hoffstein, Whyte
qTESLA	•		**Bindel**, Akleylek, Alkim, Barreto, Buchmann, Eaton, Gutoski, Krämer, ...
Round2	•		**Garcia-Morchon**, Zhang, Bhattacharya, Rietman, Tolhuizen, Torre-Arce
SABER	•		**D'Anvers**, Karmakar, Roy, Vercauteren
Three Bears	•		**Hamburg**
Titanium	•		**Steinfeld**, Sakzad, Zhao
Totals:	**21**	**3**	**Total: 24 proposals estimate SVP hardness with sieving/enumeration**

*Not included in this overview: Compact LWE, DRS, Mersenne, Odd Manhattan, Ramstake, ...

SVP hardness

Overview

Problem: How hard is SVP in high dimensions?

- Two main approaches: *enumeration* and *sieving*
  - Enumeration: memory-efficient, asymptotically slow
  - Sieving: memory-intensive, asymptotically fast
- Theoretically (large n): sieving > enumeration
- Practically (small n): enumeration > sieving
- NIST submissions: (mostly) sieving

SVP hardness

Overview

Problem: How hard is SVP in high dimensions?

- Two main approaches: *enumeration* and *sieving*
  - Enumeration: memory-efficient, asymptotically slow
  - Sieving: memory-intensive, asymptotically fast
- Theoretically (large n): sieving > enumeration
- Practically (small n): enumeration > sieving
- NIST submissions: (mostly) sieving

Problem: Can sieving still be improved?

SVP hardness

Overview

Problem: How hard is SVP in high dimensions?

- Two main approaches: *enumeration* and *sieving*
  - Enumeration: memory-efficient, asymptotically slow
  - Sieving: memory-intensive, asymptotically fast
- Theoretically (large n): sieving > enumeration
- Practically (small n): enumeration > sieving
- NIST submissions: (mostly) sieving

Problem: Can sieving still be improved?

- Theoretically: Probably not... [BDGL16, ALRW17, HKL18]

SVP hardness

Overview

Problem: How hard is SVP in high dimensions?

- Two main approaches: *enumeration* and *sieving*
  - Enumeration: memory-efficient, asymptotically slow
  - Sieving: memory-intensive, asymptotically fast
- Theoretically (large n): sieving > enumeration
- Practically (small n): enumeration > sieving
- NIST submissions: (mostly) sieving

Problem: Can sieving still be improved?

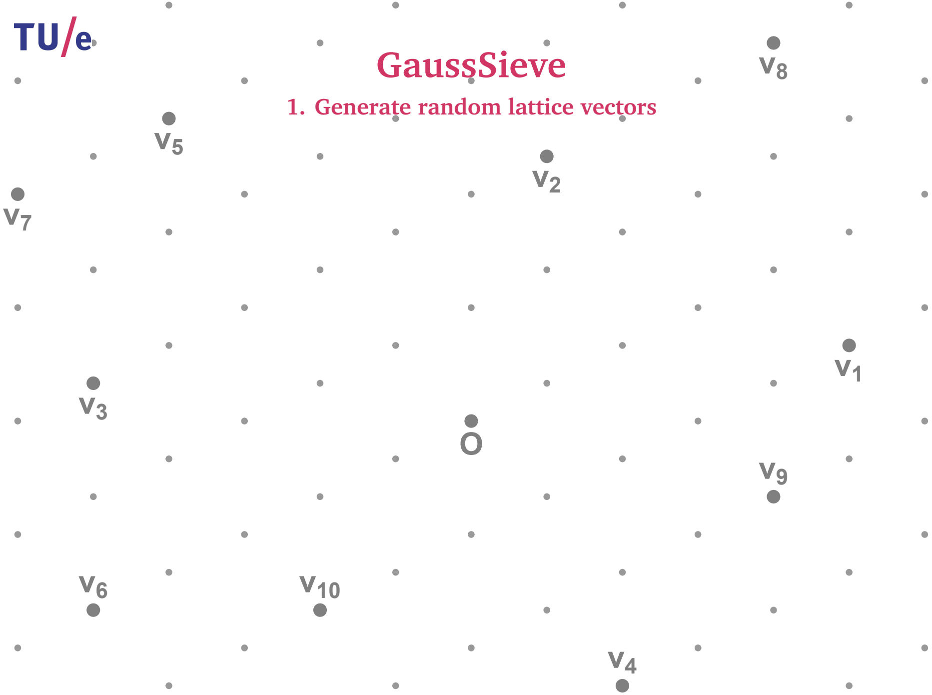
- Theoretically: Probably not... [BDGL16, ALRW17, HKL18]
- Practically: Yes! **(this work)**, [Duc18]

GaussSieve

1. Generate random lattice vectors

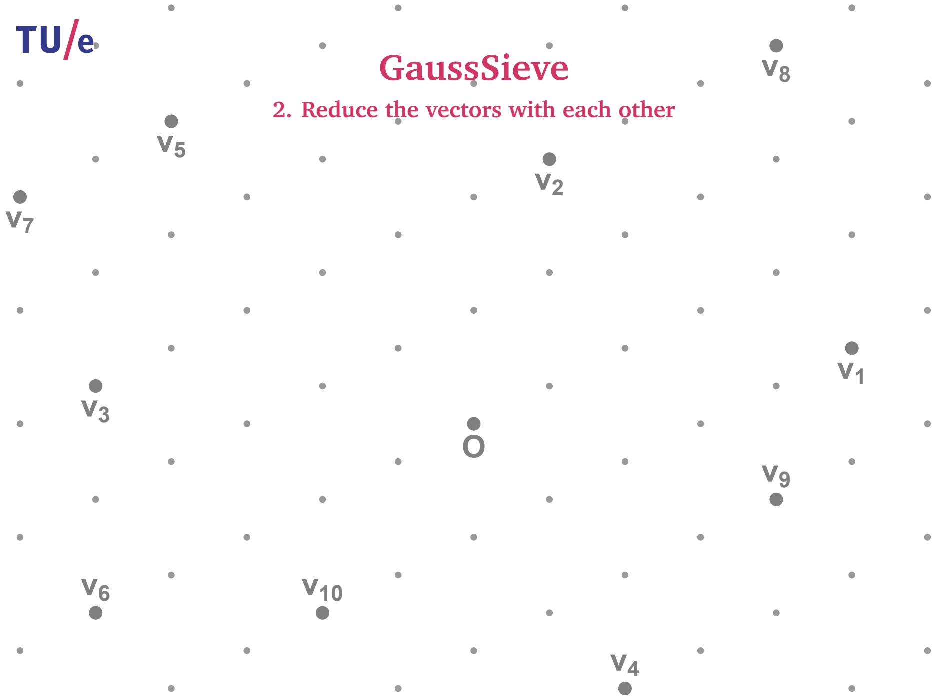
GaussSieve

1. Generate random lattice vectors



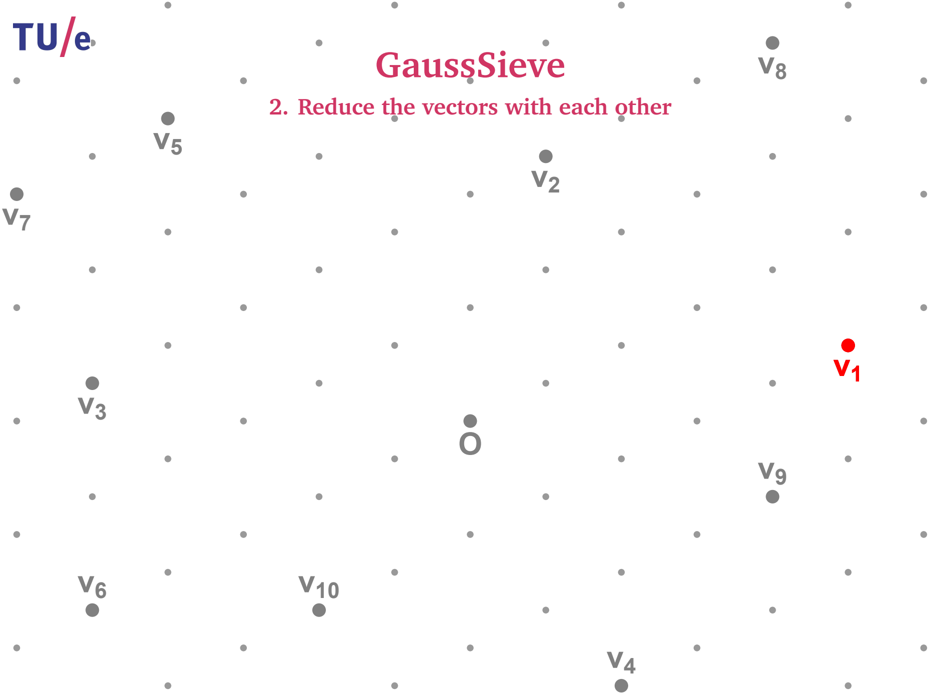
GaussSieve

2. Reduce the vectors with each other



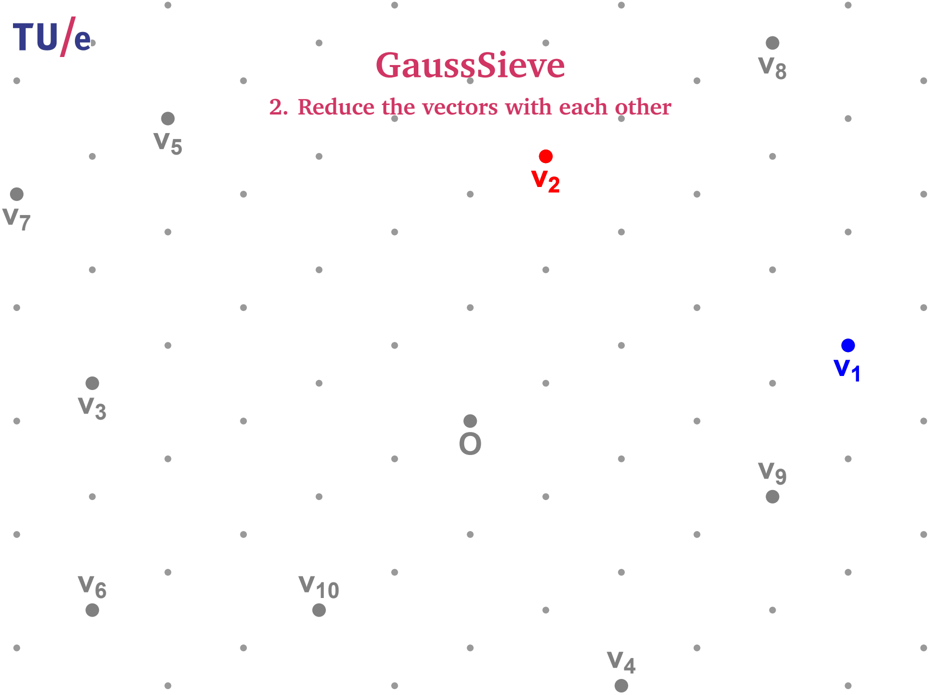
GaussSieve

2. Reduce the vectors with each other



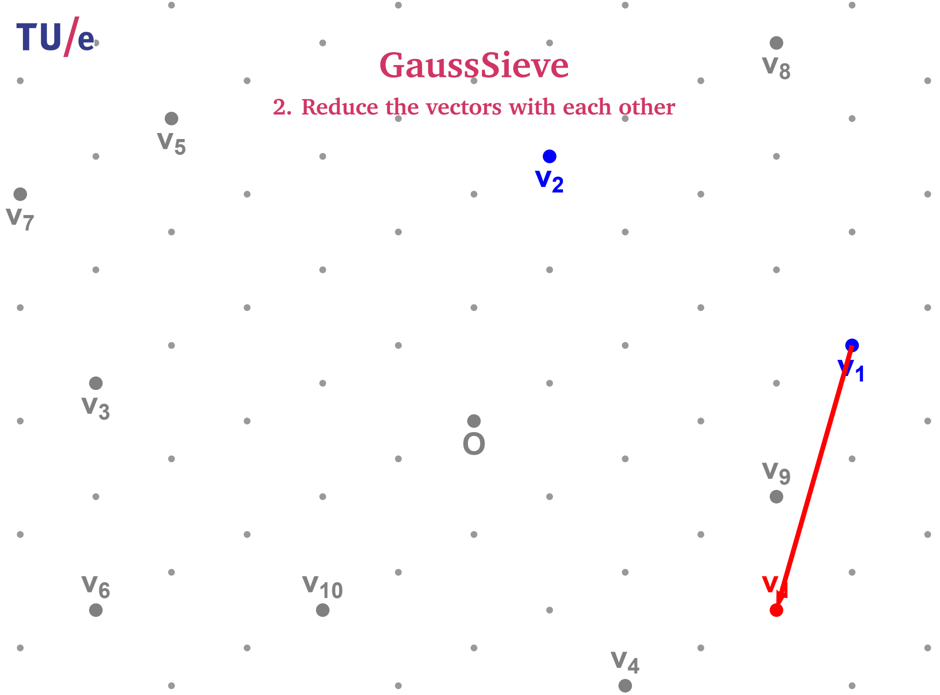
GaussSieve

2. Reduce the vectors with each other



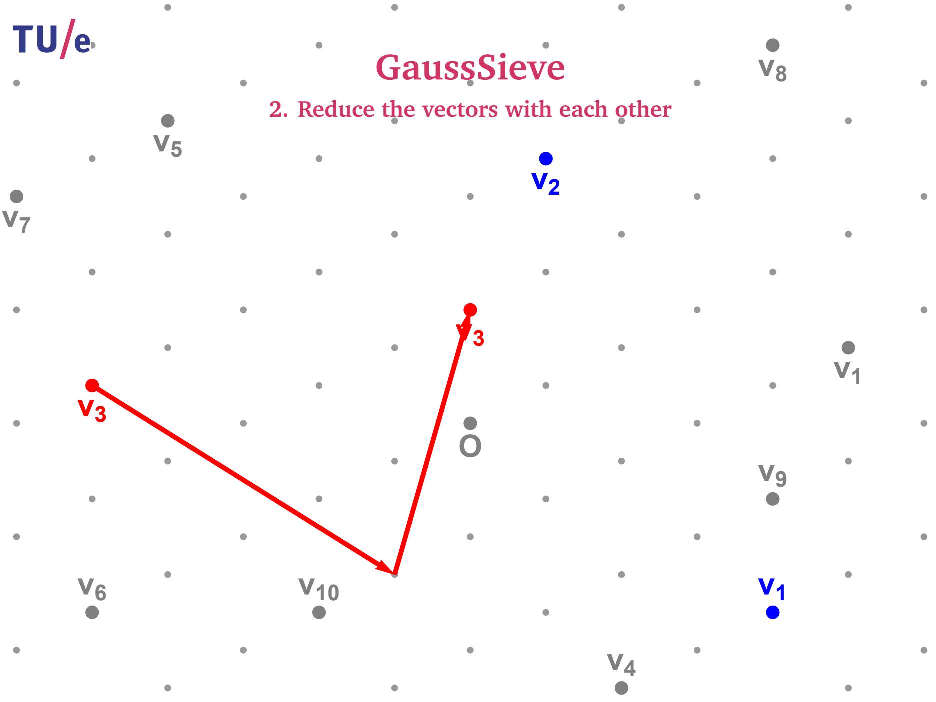
GaussSieve

2. Reduce the vectors with each other



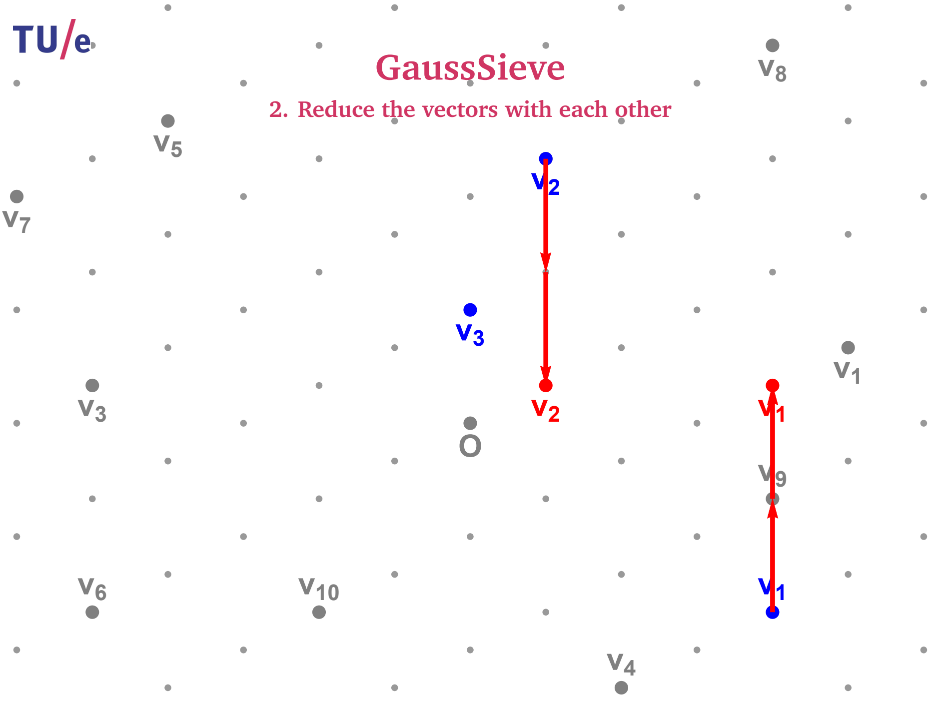
GaussSieve

2. Reduce the vectors with each other



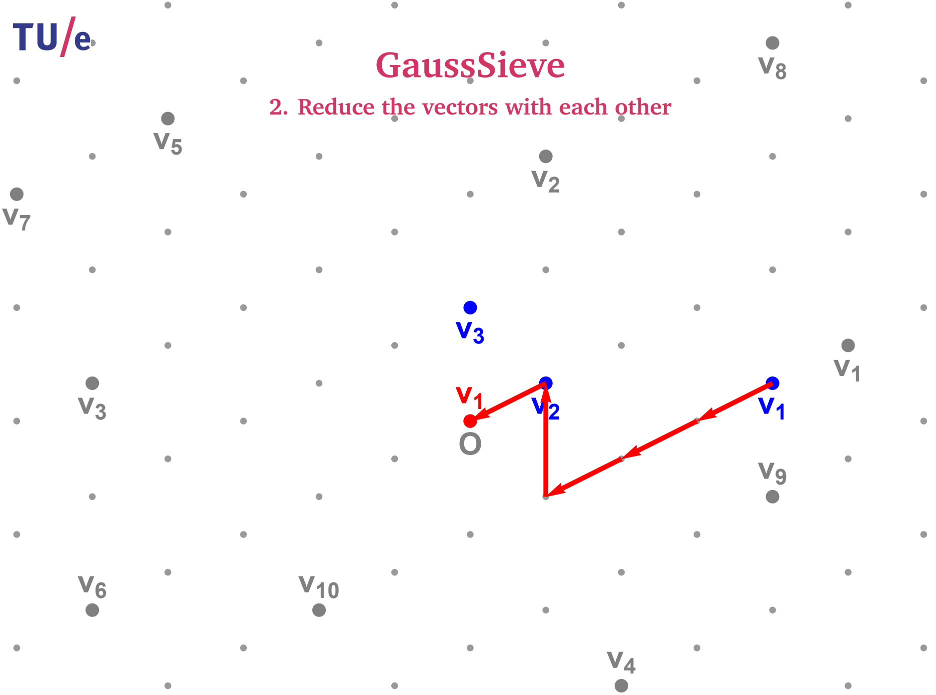
GaussSieve

2. Reduce the vectors with each other



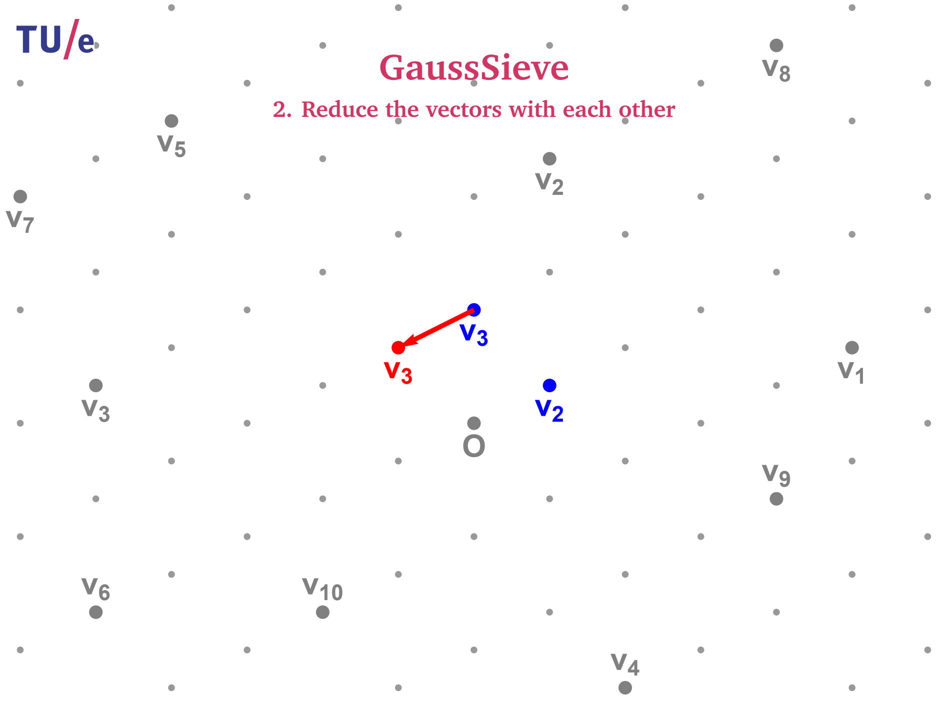
GaussSieve

2. Reduce the vectors with each other



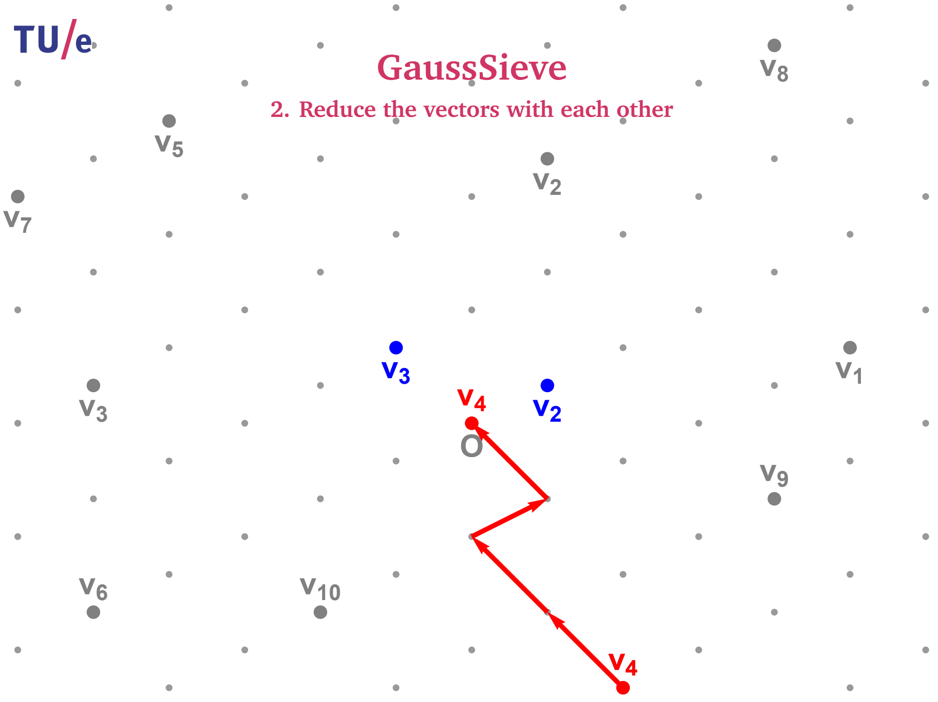
GaussSieve

2. Reduce the vectors with each other



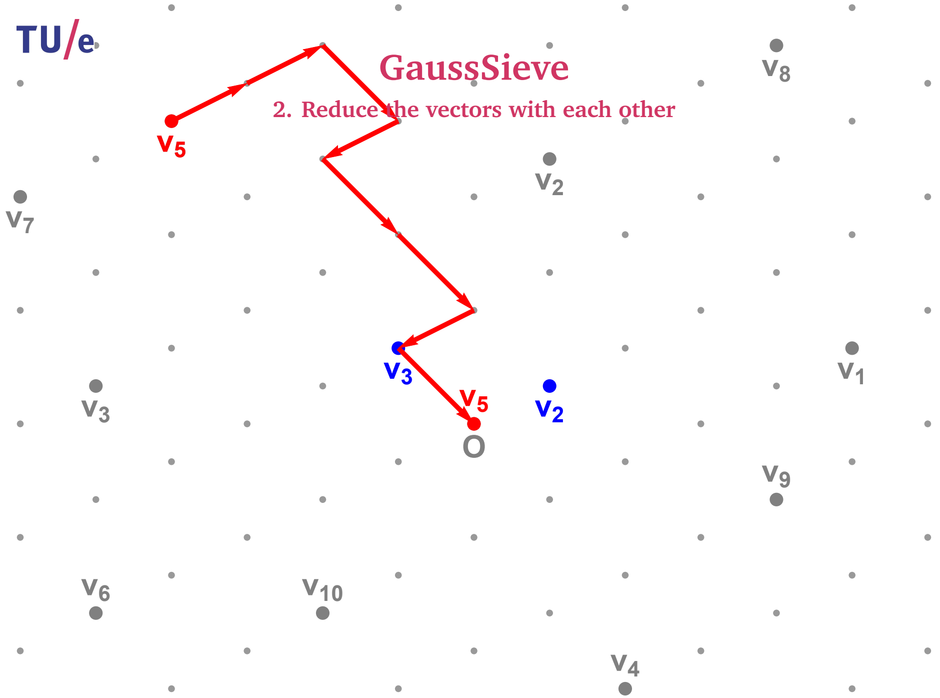
GaussSieve

2. Reduce the vectors with each other



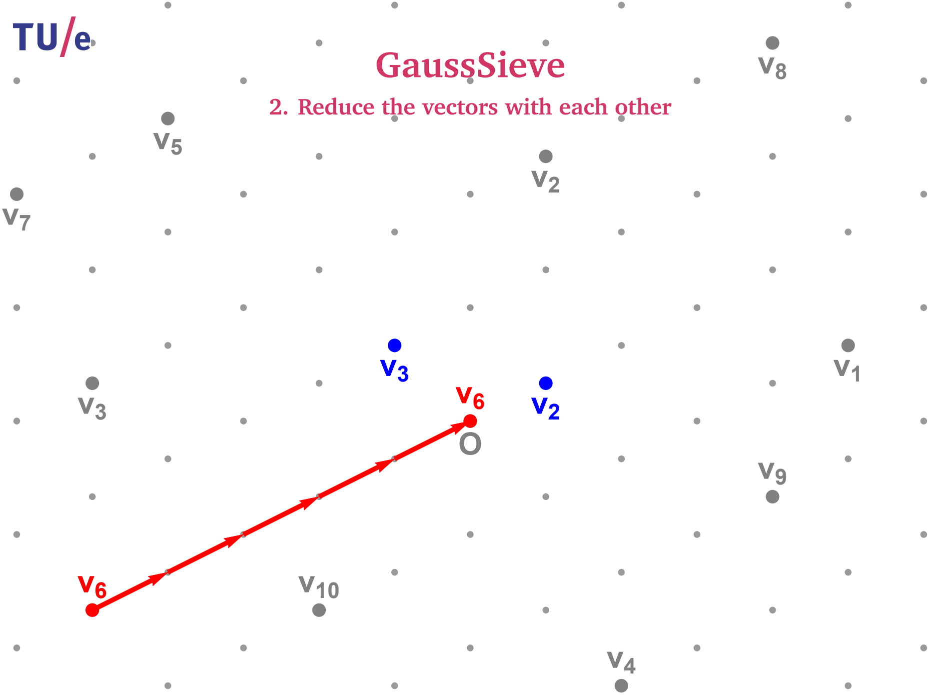
GaussSieve

2. Reduce the vectors with each other



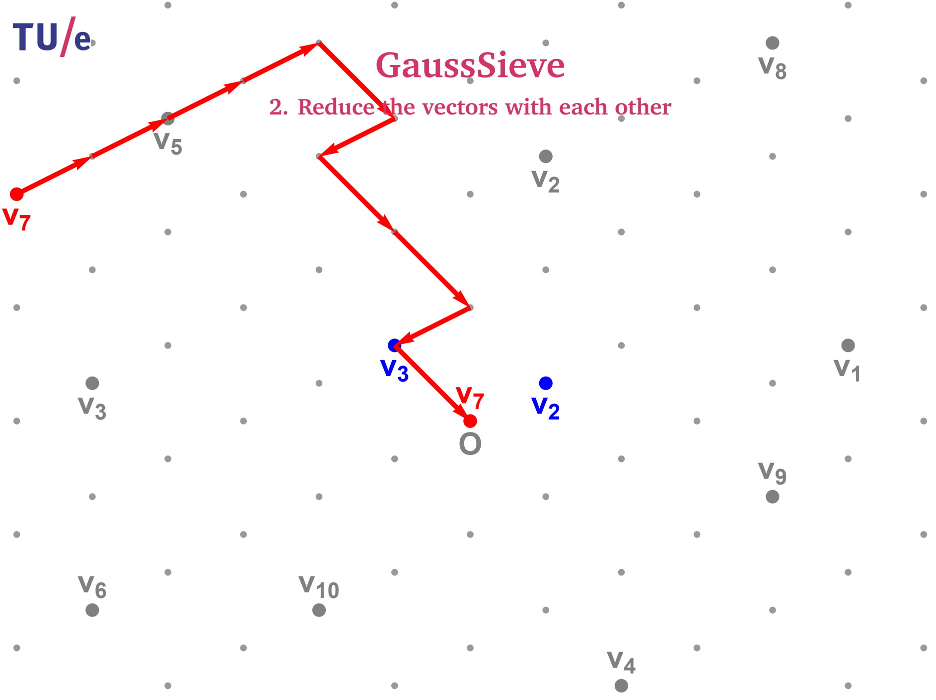
GaussSieve

2. Reduce the vectors with each other



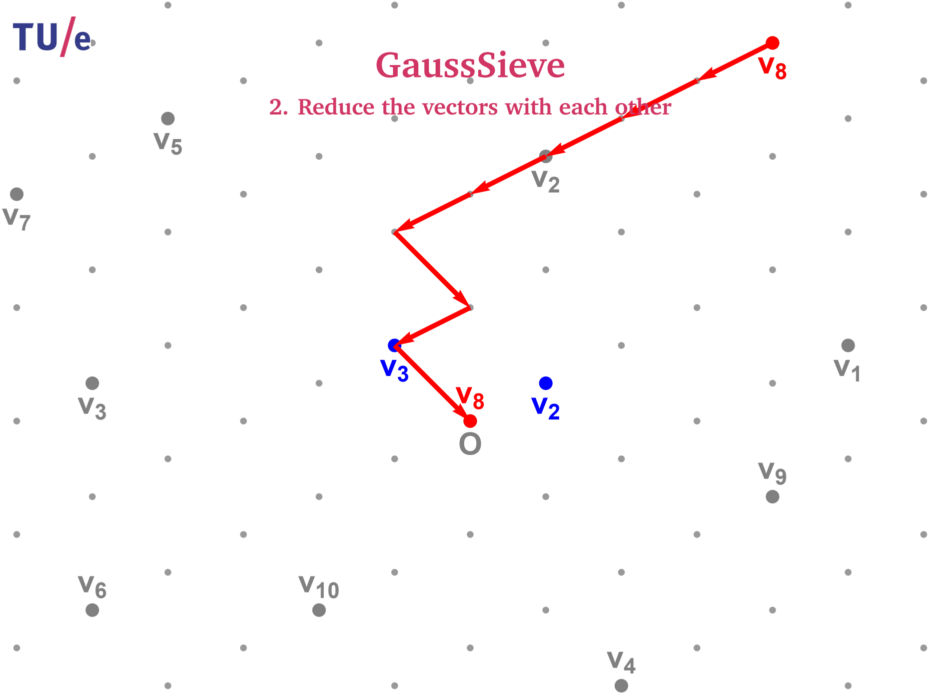
GaussSieve

2. Reduce the vectors with each other



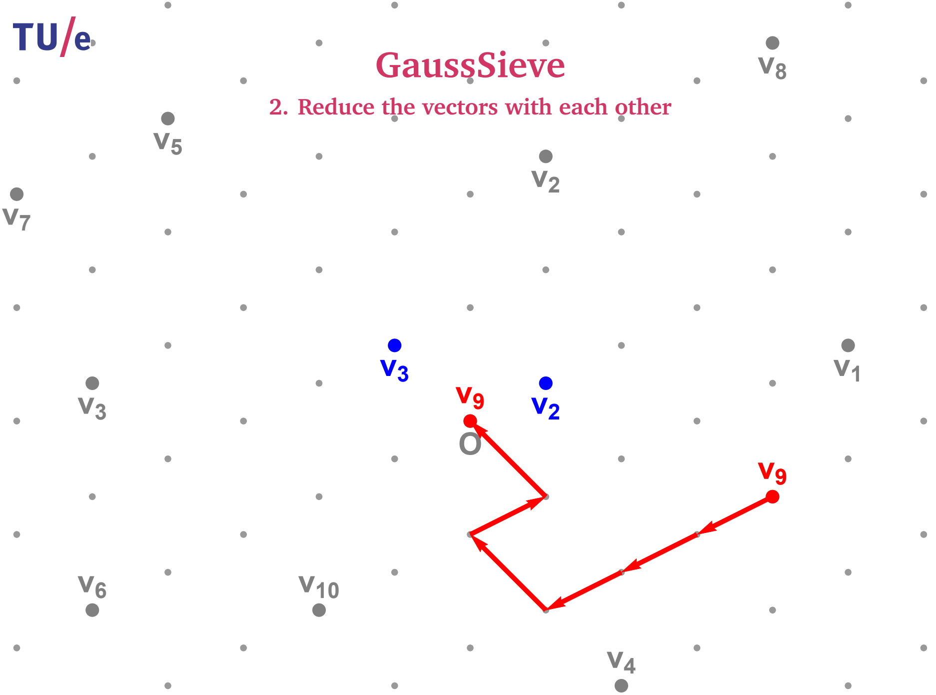
GaussSieve

2. Reduce the vectors with each other



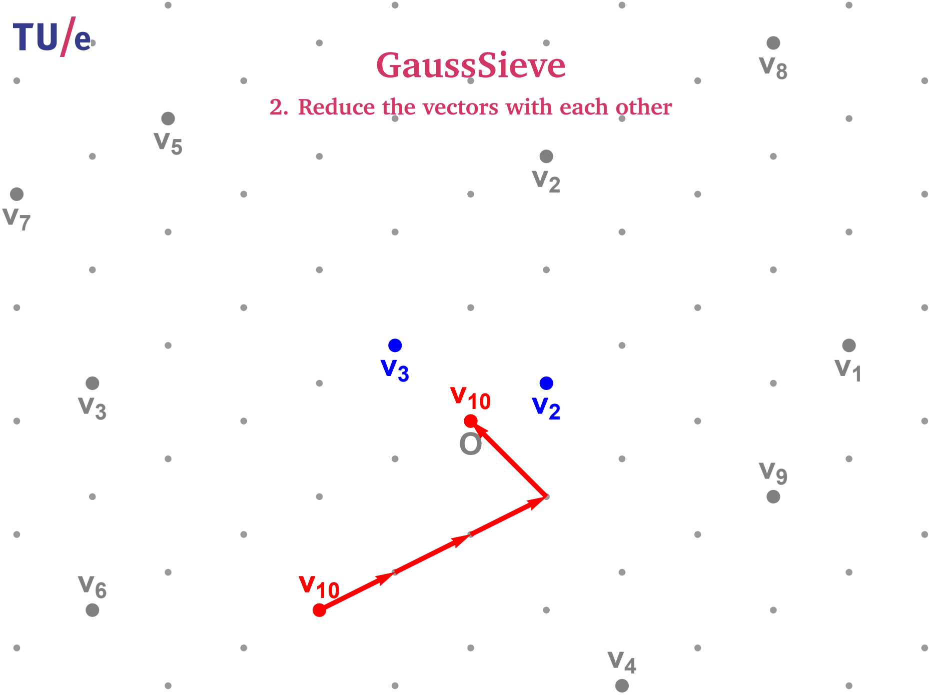
GaussSieve

2. Reduce the vectors with each other



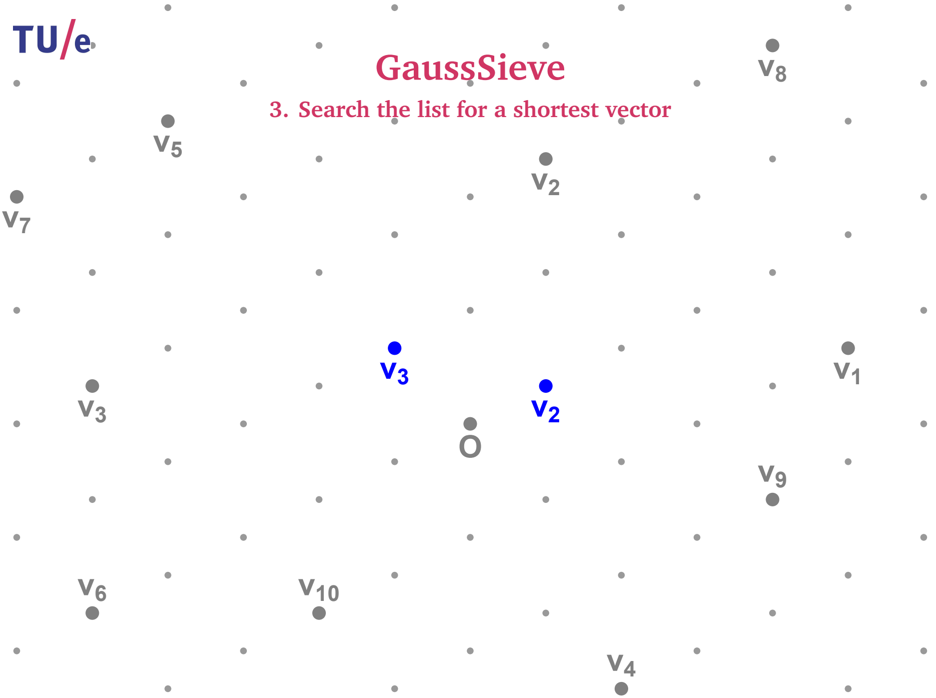
GaussSieve

2. Reduce the vectors with each other



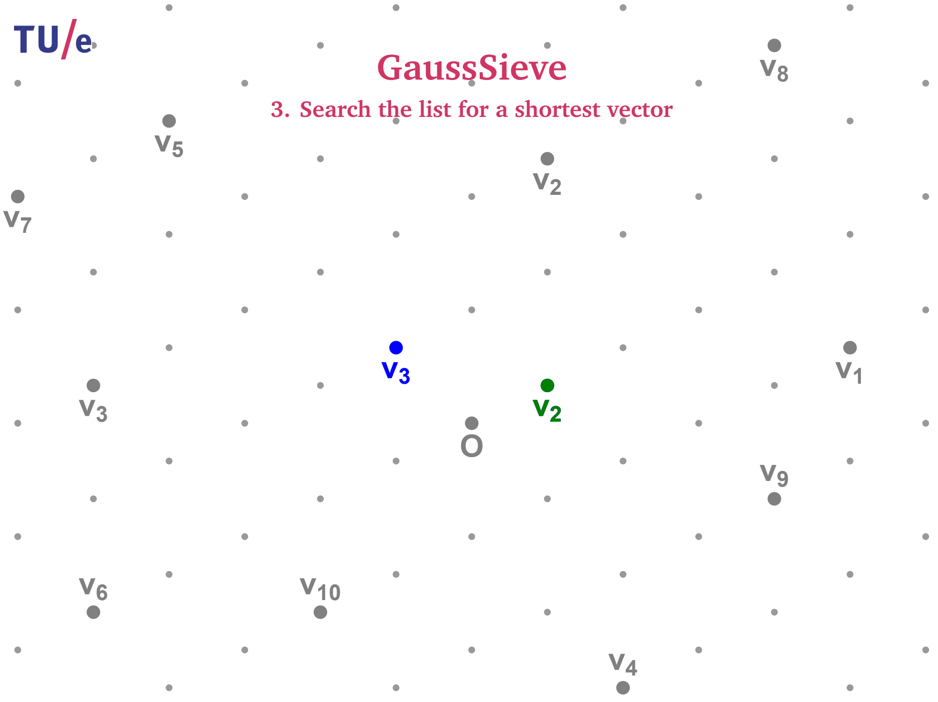
GaussSieve

3. Search the list for a shortest vector



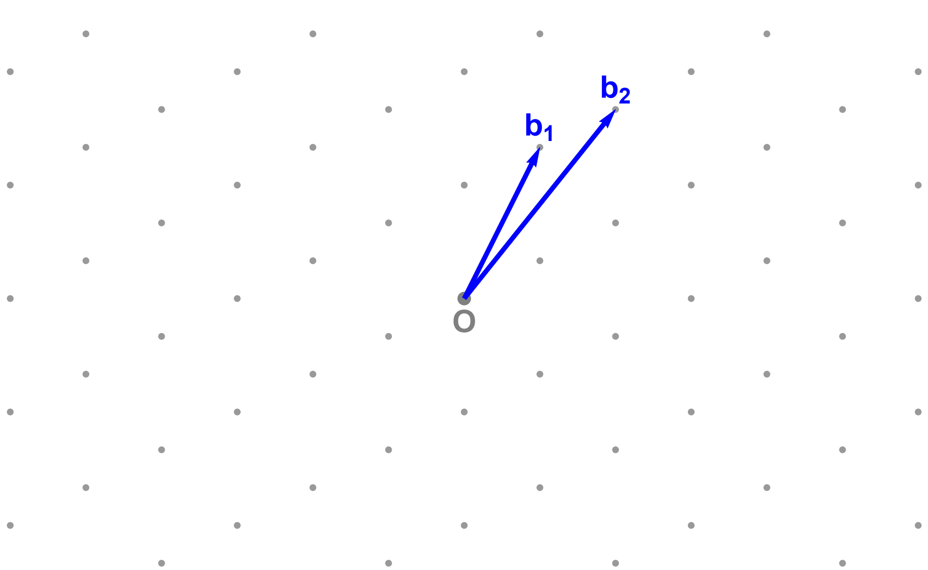
GaussSieve

3. Search the list for a shortest vector



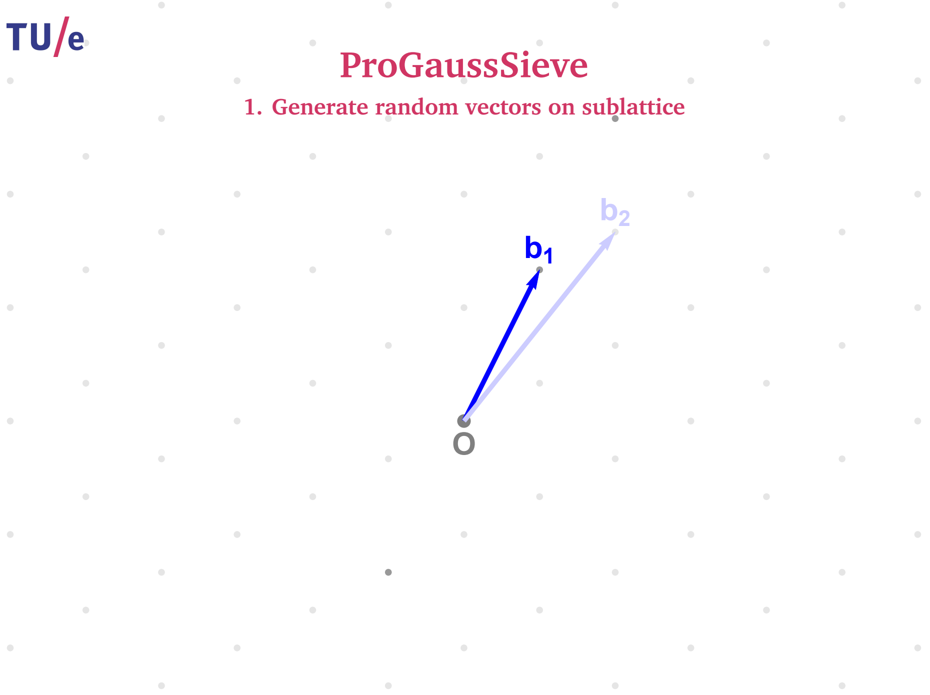
ProGaussSieve

1. Generate random vectors on sublattice



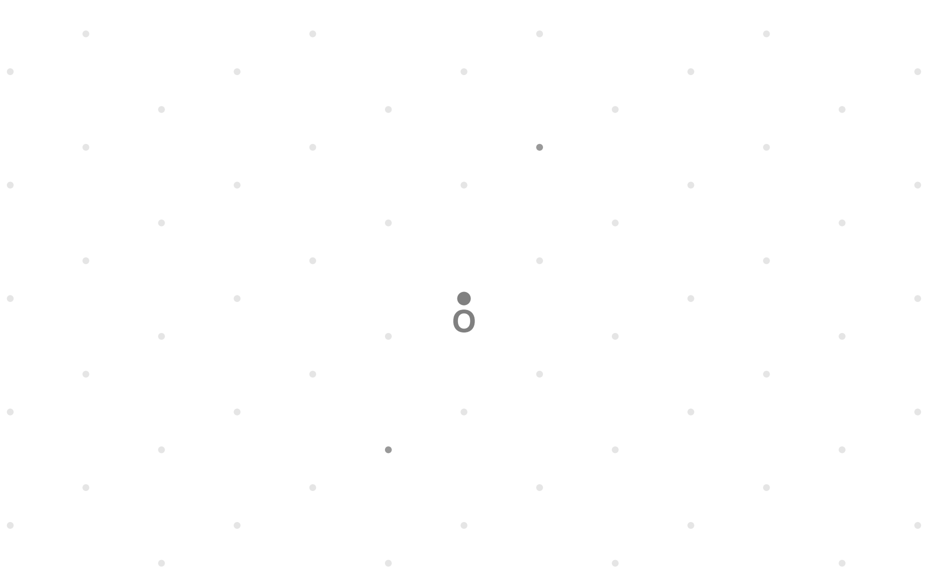
ProGaussSieve

1. Generate random vectors on sublattice



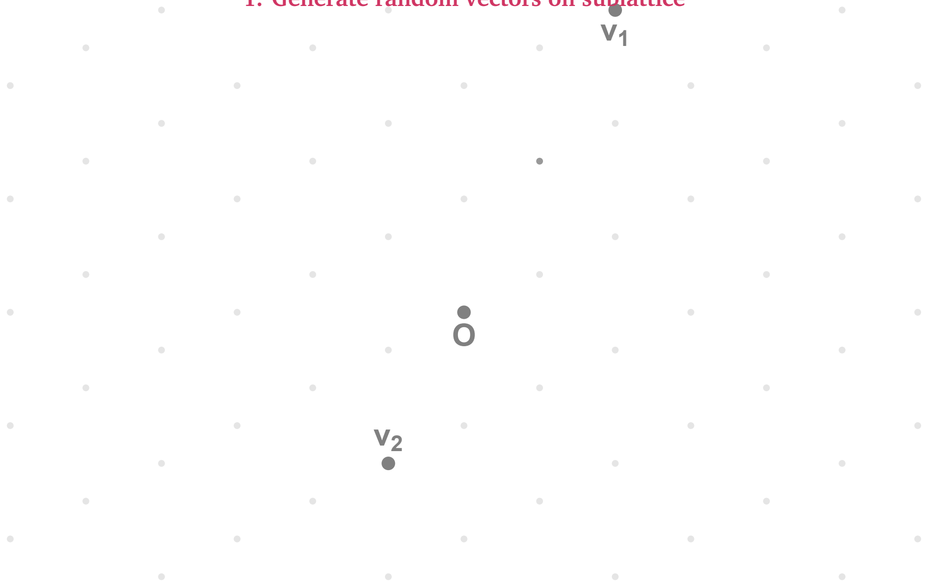
ProGaussSieve

1. Generate random vectors on sublattice



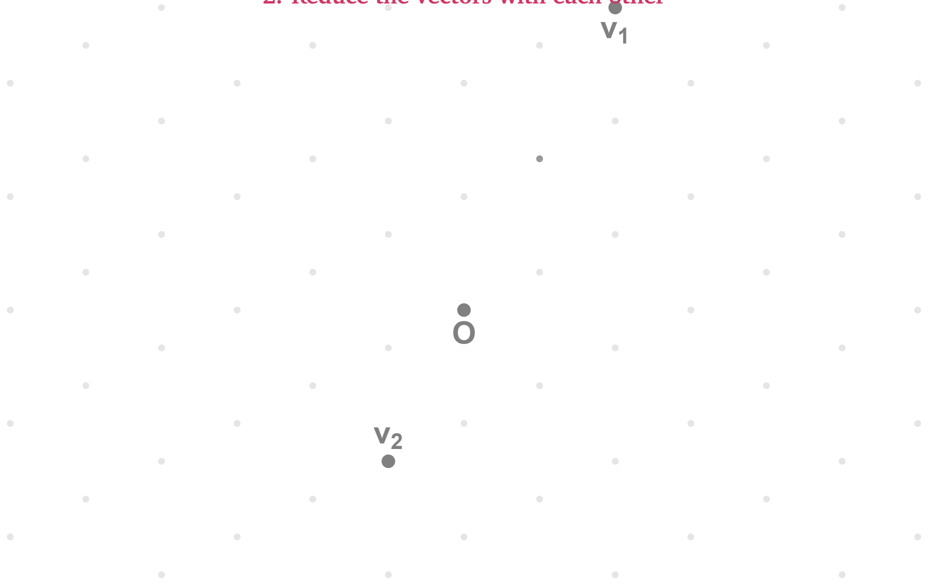
ProGaussSieve

1. Generate random vectors on sublattice



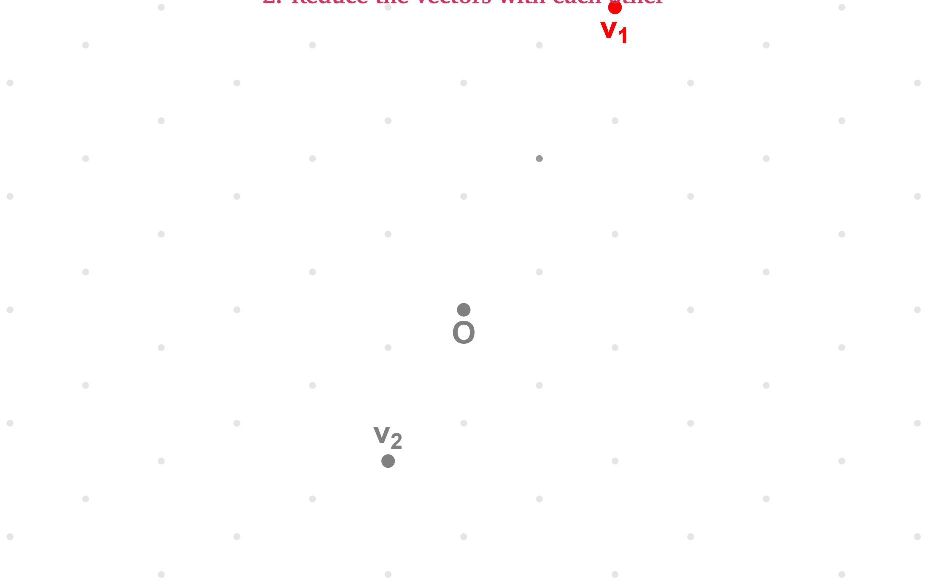
ProGaussSieve

2. Reduce the vectors with each other



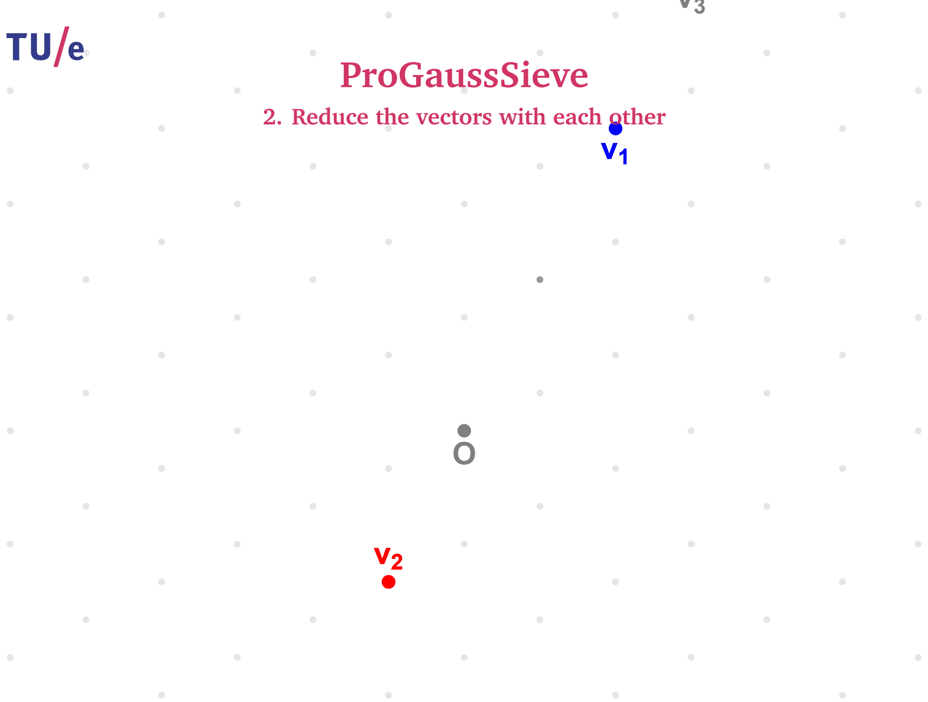
ProGaussSieve

2. Reduce the vectors with each other



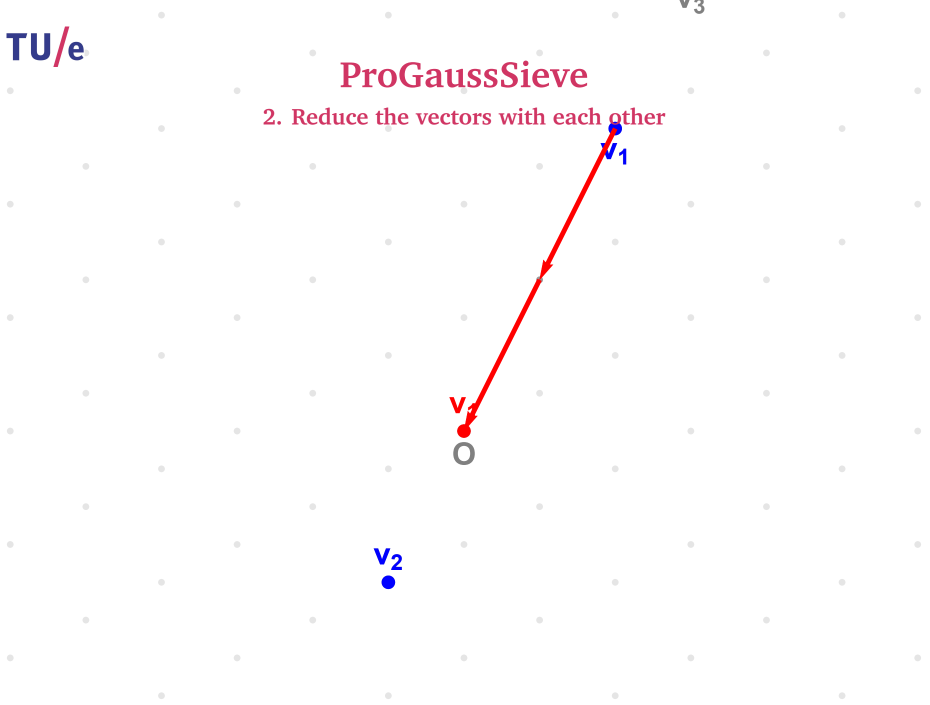
ProGaussSieve

2. Reduce the vectors with each other



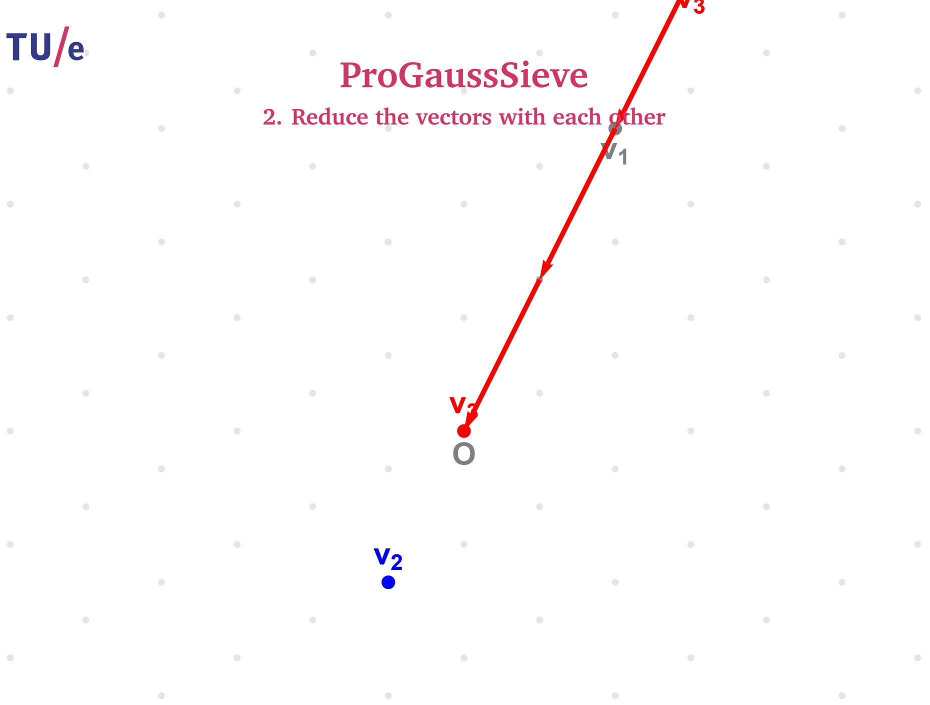
ProGaussSieve

2. Reduce the vectors with each other



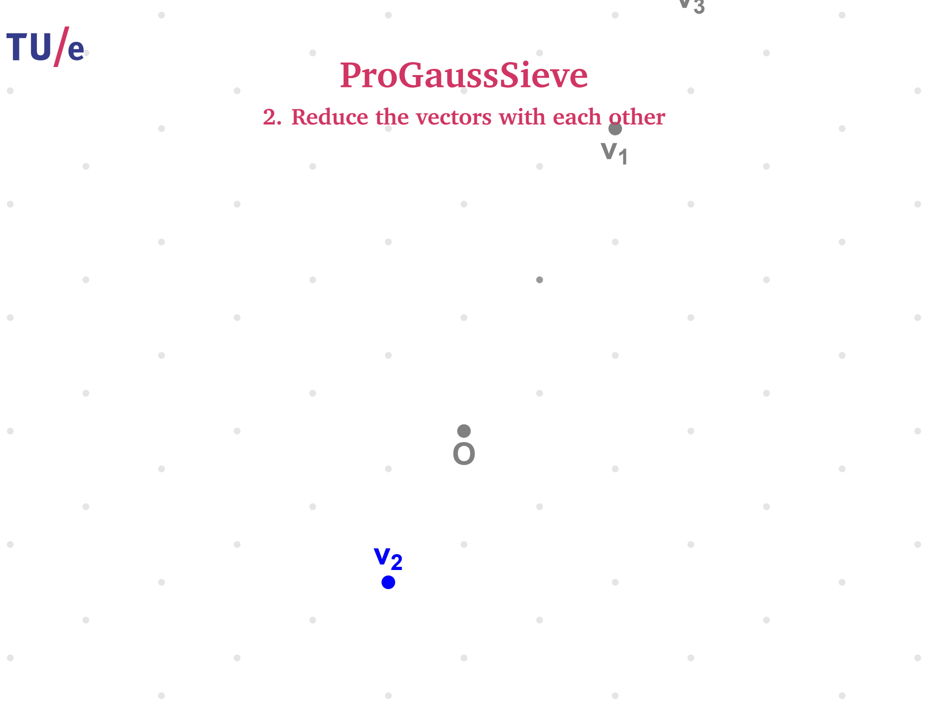
ProGaussSieve

2. Reduce the vectors with each other



ProGaussSieve

2. Reduce the vectors with each other

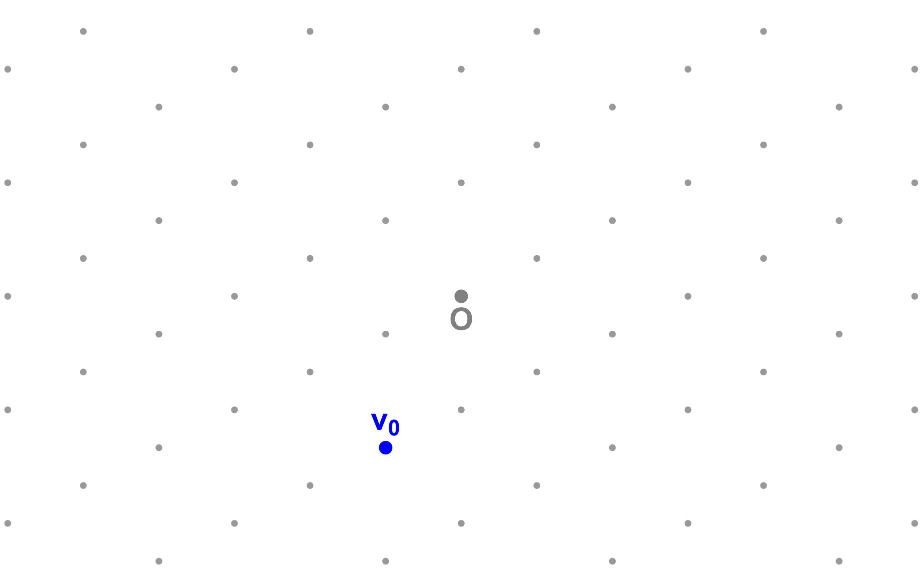


ProGaussSieve

2. Reduce the vectors with each other

ProGaussSieve

3. Generate random vectors on full lattice

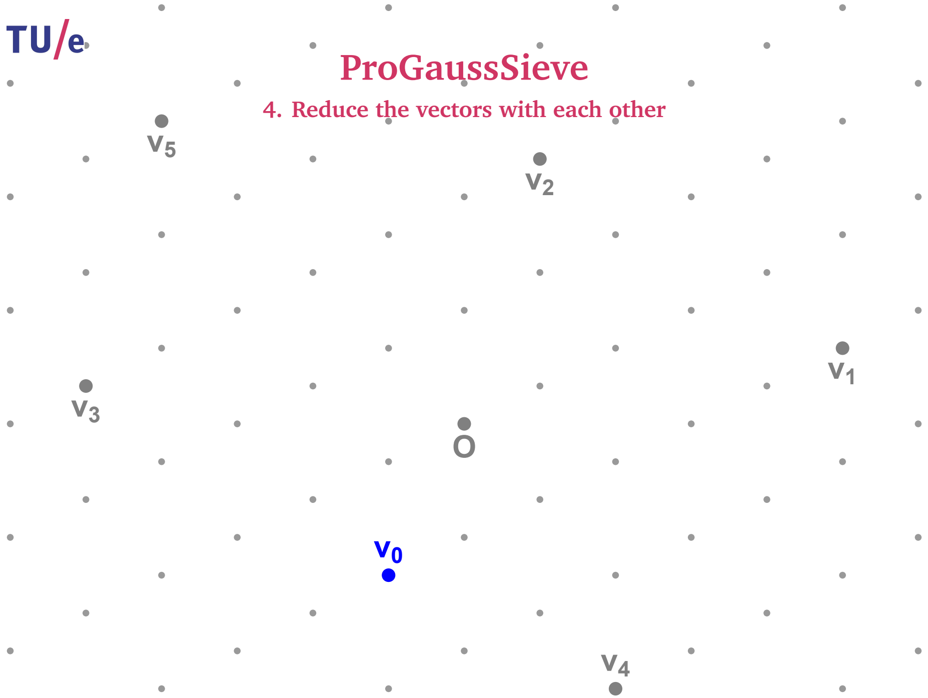


ProGaussSieve

3. Generate random vectors on full lattice

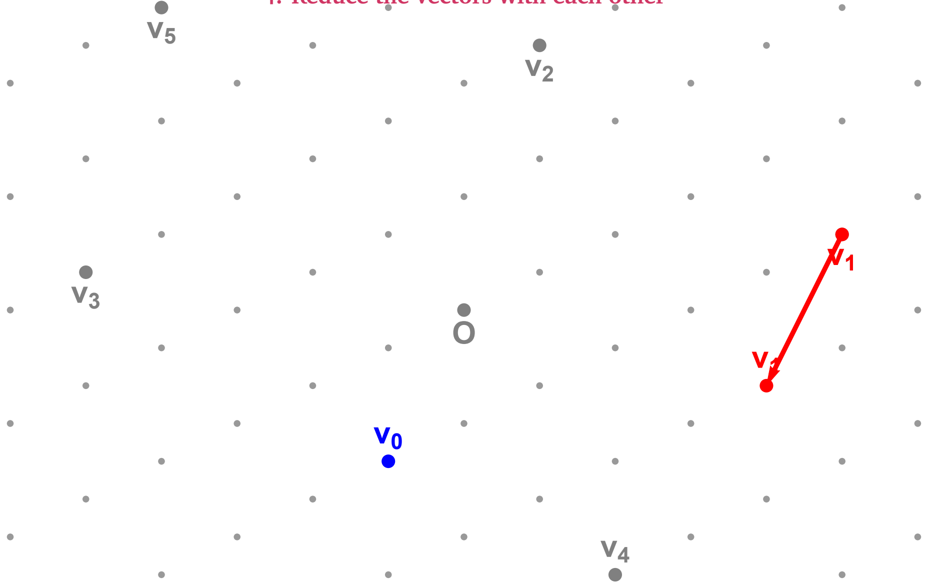
ProGaussSieve

4. Reduce the vectors with each other



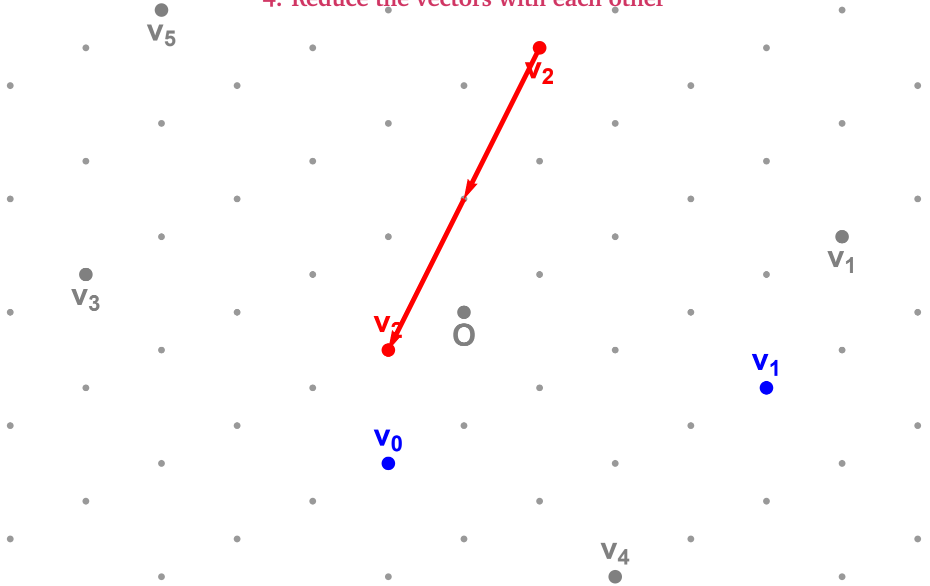
ProGaussSieve

4. Reduce the vectors with each other



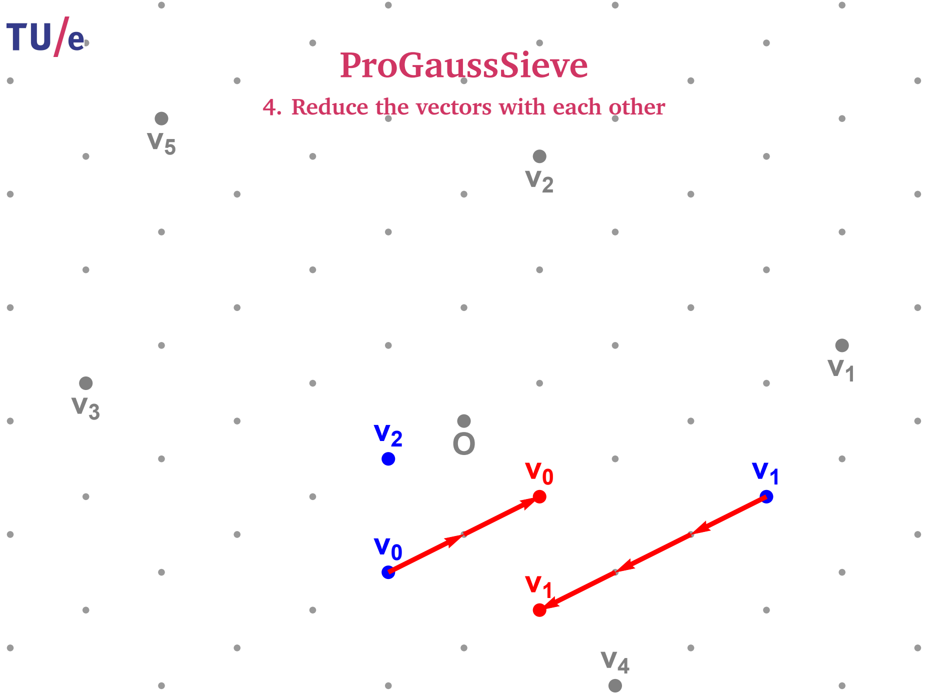
ProGaussSieve

4. Reduce the vectors with each other



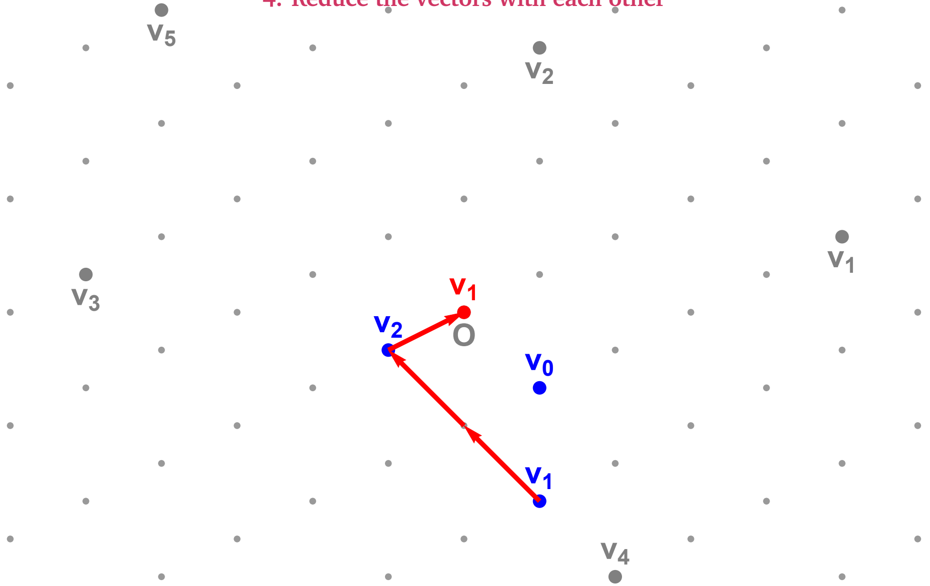
ProGaussSieve

4. Reduce the vectors with each other



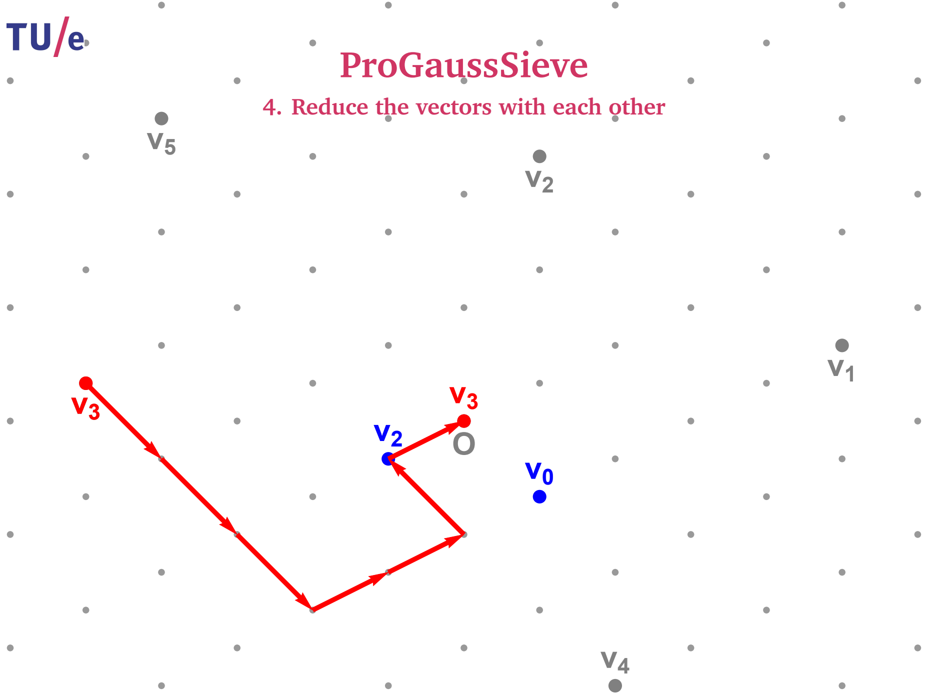
ProGaussSieve

4. Reduce the vectors with each other



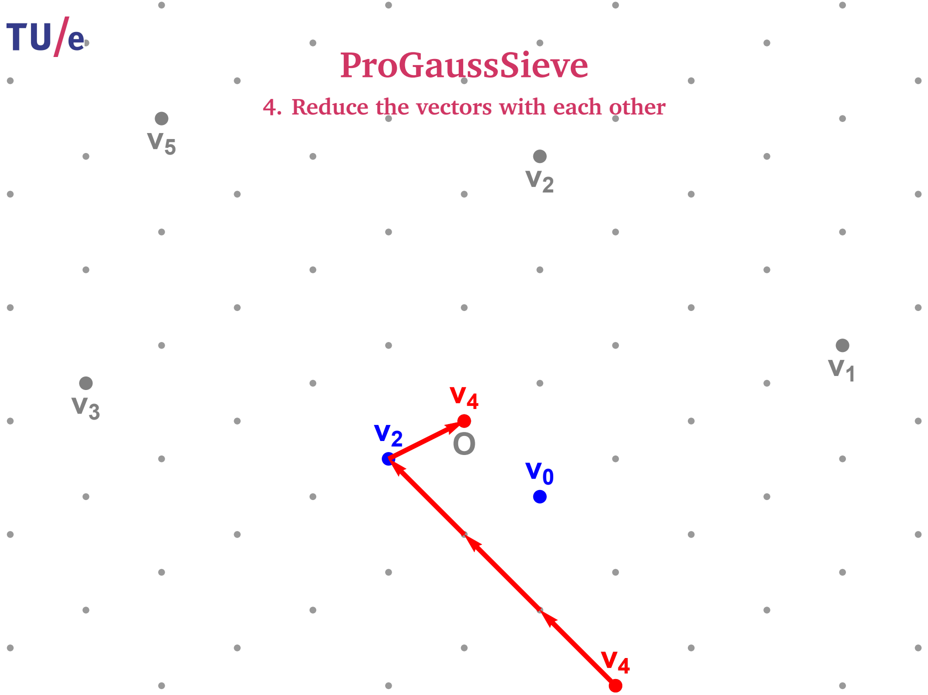
ProGaussSieve

4. Reduce the vectors with each other



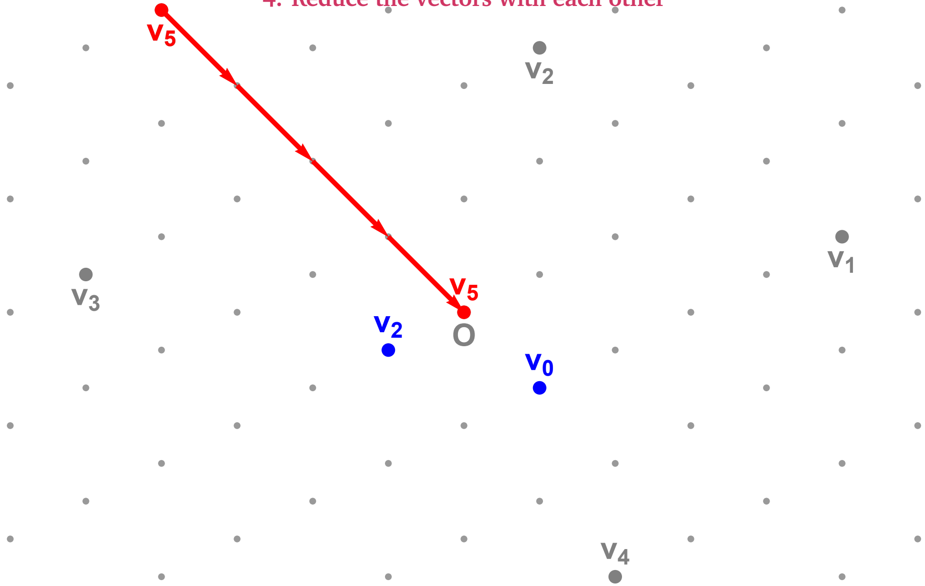
ProGaussSieve

4. Reduce the vectors with each other



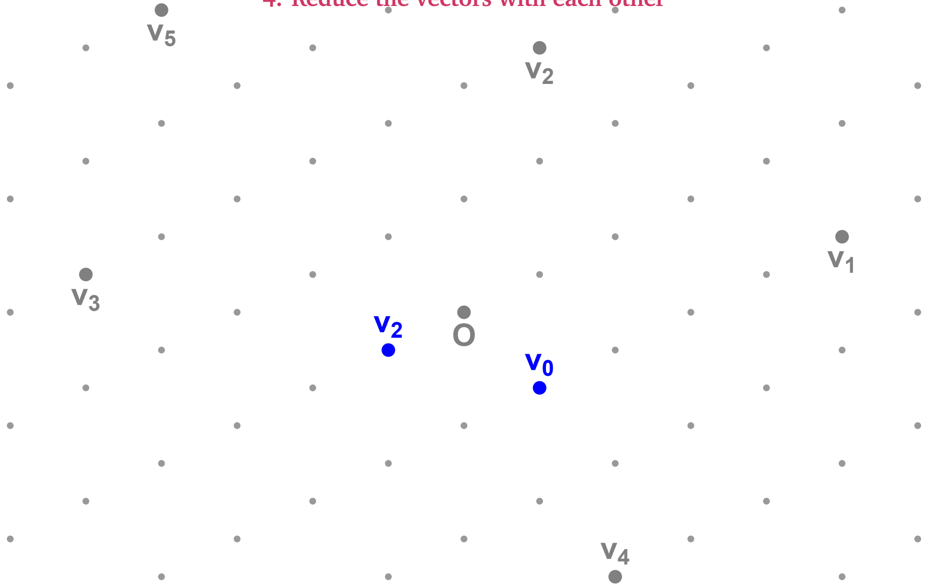
ProGaussSieve

4. Reduce the vectors with each other



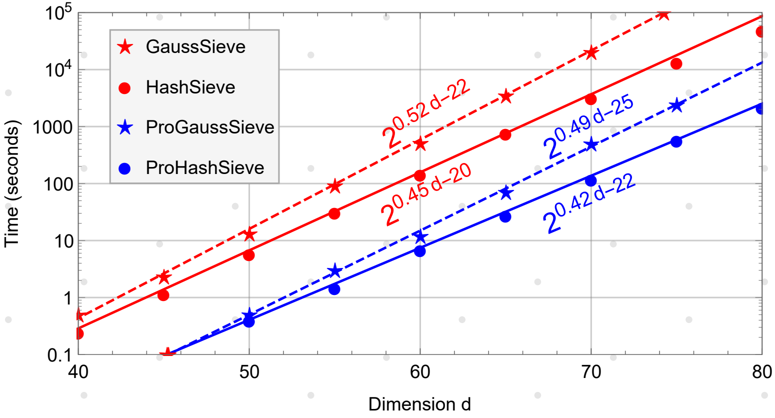
ProGaussSieve

4. Reduce the vectors with each other



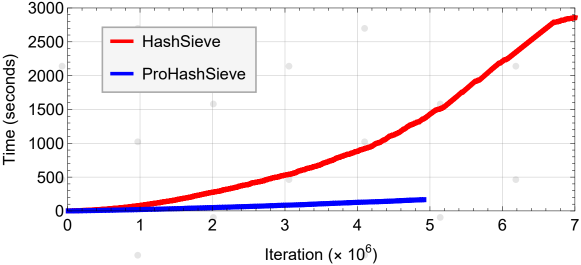
Progressive sieving

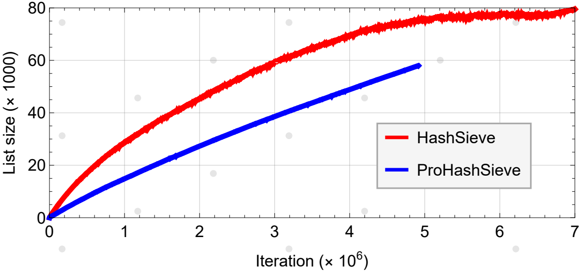
Time complexities



Progressive sieving

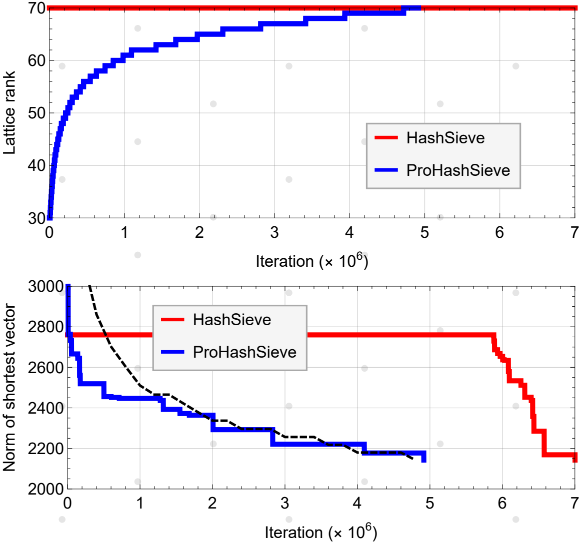
Execution profiles ($n = 70$)





Progressive sieving

Execution profiles ($n = 70$)



Progressive sieving

Effects of basis reduction ($n = 70$)

Exact SVP	⟵ **GaussSieve** ⟶			⟵ **HashSieve** ⟶		
	LLL	BKZ-10	BKZ-30	LLL	BKZ-10	BKZ-30
Standard sieving	19100	18100	16500	3300	3050	2900
Progressive sieving	595	440	390	165	125	115
Speedup factor	**32**×	**41**×	**42**×	**20**×	**24**×	**25**×

Approximate SVP	⟵ **GaussSieve** ⟶			⟵ **HashSieve** ⟶		
($\gamma = \mathbf{1.1}$)	LLL	BKZ-10	BKZ-30	LLL	BKZ-10	BKZ-30
Standard sieving	18500	17200	15600	3180	2960	2700
Progressive sieving	120	40	3	65	20	2
Speedup factor	**150**×	**400**×	**5000**×	**50**×	**150**×	**1000**×

Conclusion

Progressive lattice sieving

- Uses recursive approach (rank reduction)
- Finds approximate solutions faster
- Benefits more from reduced bases
- Better predictability
- Faster, using slightly less memory
- No theoretical/asymptotic improvements...
  - Best classical time: $(3/2)^{n/2+o(n)} \approx 2^{0.292n+o(n)}$
  - Best quantum time: $(13/9)^{n/2+o(n)} \approx 2^{0.265n+o(n)}$

Questions?