Advanced Cryptography

Lattice-based Cryptography

Christophe Petit

University of Oxford

Why lattice-based cryptography ?

- Connection to NP-hard problems
- Worst-case vs average-case hardness
- No quantum attack
- Assumptions diversity : Don't put all eggs in same basket
- Faster solutions to old problems (encryption, signatures)
- First solutions to other problems
  (fully homomorphic encryption, multilinear maps)

Lattice-based cryptanalysis

- More parameters than discrete logarithms & factorization hence somewhat harder to evaluate
- Other schemes also solved by reduction to lattice problem
  - Knapsack cryptosystems
  - Factoring with partial key exposure
  - Lattice attacks on DSA, ECDSA

  (first applications of lattices in cryptography)

Outline

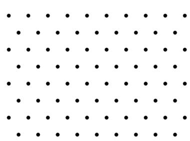
References

- Micciancio-Goldwasser, *Complexity of Lattice Problems*
- Joux, *Algorithmic cryptanalysis*
- Micciancio-Regev, *Lattice-based cryptography*
- Peikert, *A decade of lattice cryptography*

Outline

Lattices

- **Lattice** L : discrete subgroup of $\mathbb{R}^n$
  - Subgroup : L contains $av_1 + bv_2$ for all $a, b \in \mathbb{Z}$ and $v_1, v_2 \in L$
  - Discrete : non continuous ($\exists$ centered ball at 0 with no other lattice element)
- **Dimension** of L is n
- A lattice is **integer** if all lattice elements have integer coefficients



Picture source : Wikipedia

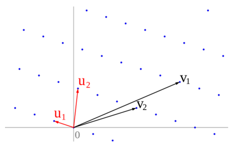
Lattices

- A **basis** of L is a minimal set of elements $\{v_i\}$ such that

$$L = \left\{ \sum_{i=1}^{r} a_i v_i | a_i \in \mathbb{Z} \right\}$$

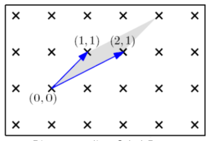
- **Rank** r of L is the size of a basis
- A lattice is **full-rank** if $r = n$
- We often represent a basis $\{v_i\}$ as a matrix $V \in \mathbb{R}^{n \times r}$, one column for all coefficients of one basis element
- In other words $L = \{Vx, x \in \mathbb{Z}^r\}$

Equivalent bases



- The red an black bases generate the same lattice : $v_1 = 2u_2 - 5u_1$, $v_2 = u_2 - 3u_1$, and $u_1 = v_1 - 2v_2$, $u_2 = 3v_1 - 5v_2$
- The sets $\{u_i\}$, $\{v_i\}$ generate the same lattice iff there exists $S \in \mathbb{Z}^{r\times r}$ such that $U = VS$ and $\det S = \pm 1$

Fundamental parallelepiped and Determinant



Picture credit : Oded Regev

- Let B be a lattice basis
- We can associate to it a **fundamental parallelepiped** $\mathcal{P}(B)$ consisting of all points modulo B
- The **determinant** of lattice L is $\det(L) = \sqrt{|\det(B \cdot B^t)|}$ (does not depend on basis B) $(= |\det B|$ if $n = r)$
- Determinant is the **volume** of fundamental parallelepiped

Scalar product and Euclidean norm

- Given $u = (u_1, \ldots, u_n), v = (v_1, \ldots, v_n) \in \mathbb{R}^n$, their **scalar product** is $\langle u, v\rangle := \sum_{i=1}^n u_i v_i$
- Scalar product is **bilinear** : $\forall \alpha \in \mathbb{R}$, $\langle \alpha u, v\rangle = \langle u, \alpha v\rangle = \alpha\langle u, v\rangle$
- $u, v \in \mathbb{R}^n$ are **orthogonal** if $\langle u, v\rangle = 0$
- **Euclidean norm** of $v \in \mathbb{R}^n$ is $||v|| = \sqrt{\sum_i v_i^2} = \sqrt{\langle v, v\rangle}$
- Basis $\{b_1, \ldots, b_n\}$ is **orthogonal** if $\langle b_i, b_j\rangle = 0 \quad \forall i \neq j$, in other words iff $B^t \cdot B$ is a diagonal matrix
- $u, v \in \mathbb{R}^n$ are **parallel** if $\langle u, v\rangle = ||u|| \cdot ||v||$

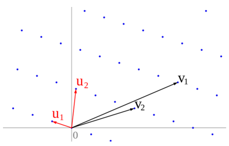
The shortest vector problem (SVP)

- We call λ_1 the shortest norm in the lattice

$$\lambda_1(L) = \min_{v\in L,\ v\neq 0} ||v||$$

- **Shortest vector problem** (SVP) : given a basis $\{v_1, \ldots, v_n\}$ for L, find $v \in L$ with $||v|| = \lambda_1(L)$

Good and bad bases



- Some bases make SVP easier
- A "good" basis has shorter vector norms
- A "good" basis has nearly orthogonal vectors
  (as nearly parallel vectors can lead to shorter vectors)

Upper bounding shortest vectors (1)

- Convex body theorem : For any lattice L of rank n, any convex set $S \subset span(L)$ symmetric about the origin, if $vol(S) > 2^n \det L$ then S contains nonzero lattice point

Proof :

- Consider a fundamental parallelipiped $\mathcal{P}(B)$ consisting of all points modulo a basis B of L
- Consider the set $S' = \{x \mid 2x \in S\}$
- By volume condition there exist $z_1, z_2 \in S'$ reducing to same point in $\mathcal{P}(B)$, i.e. $z_1 - z_2 \in L$
- By definition $2z_1, 2z_2 \in S$ and since S symmetric and convex we have $z_1 - z_2 \in S$

Upper bounding shortest vectors (2)

- Minkowski's first theorem : we have

$$\lambda_1 < \sqrt{n}(\det L)^{1/n}$$

Proof : remark that volume of ball $\mathcal{B}(0, r)$ is bigger than $(2r/\sqrt{n})^n$ and apply previous theorem on $S = \mathcal{B}(0, \sqrt{n}(\det L)^{1/n})$

- Minkowski's second theorem : we have

$$\left(\prod_{i=1}^{n} \lambda_i\right)^{1/n} < \sqrt{n}(\det L)^{1/n}$$

where the **successive minima** $\lambda_k(L)$ are the smallest λ such that there are at least k linearly independent vectors with norms at most λ (proof : see Goldwasser-Micciancio)

Expected size of shortest vector

- **Gaussian heuristic** : let $V = \det(L)$.
  If L is a reasonably random lattice we expect that

$$\lambda_1 \approx \text{ radius of a ball with volume } V$$

  (only a factor 2 smaller than Minkowski's bound)
- For Euclidean norm we have $V(\mathcal{B}(0, R)) = \frac{\pi^{n/2}}{(n/2)!} R^n$
- This heuristic works well for many cryptographic lattices
- Some crypto lattice distributions have very small λ_1 by construction ; then use similar heuristic for other λ_i

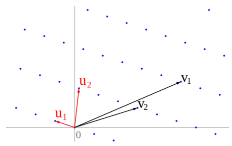
The closest vector problem (CVP)

- For a lattice L and a point $t \in \mathbb{R}^n$, define distance

$$d(t, L) := \min_{v \in L} ||v - t||$$

- **Closest vector problem** :
  Given a basis $\{v_1, \ldots, v_n\}$ for L and given $t \in \mathbb{R}^n$, find $v \in L$ with $||v|| = d(t, L)$

Good and bad bases



- Good bases also make CVP easier : all points in the fundamental parallelepiped are close to basis vectors
- See later Babai's nearest plane algorithm

Decisional SVP and CVP

- **Decision-SVP :** Given a basis $\{v_1, \ldots, v_n\}$ for L and a rational $r \in \mathbb{Q}$, determine whether $\lambda_1(L) \leq r$ or not
- **Decision-CVP :** Given a basis $\{v_1, \ldots, v_n\}$ for L, a point $t \in \mathbb{Z}^n$ and a rational $r \in \mathbb{Q}$, determine whether $d(t, L) \leq r$ or not
- Can solve decision problems if can solve search problems
- Converse also true, but needs some work (see later)

Are SVP and CVP hard ?

- Decisional CVP is NP-hard
- Search and Decisional CVP are equivalent
- Search and Decisional SVP are equivalent
- Can solve SVP if can solve CVP
- Heuristically the converse if also true

- See later !

Approximate SVP and CVP

- γ**-approximate shortest vector problem** : Given a basis $\{v_1, \ldots, v_n\}$ for L, find $v \in L$ with $||v|| \leq \gamma\lambda_1(L)$
- γ**-approximate closest vector problem** : Given a basis $\{v_1, \ldots, v_n\}$ for L and given $t \in \mathbb{R}^n$, find $v \in L$ with $||v|| \leq \gamma d(t, L)$
- Standard SVP and CVP if $\gamma = 1$

Are approximate SVP and CVP hard ?

- Still NP-hard for $\gamma < n^{1/\log\log n}$
- Becomes easier for larger γ
- Unlikely to be NP-hard for $\gamma > \sqrt{n/\log n}$
- LLL achieves $\gamma = 2^{(n-1)/2}$ in polynomial time (see later)
- In cryptography we need $\gamma = n^c$ hard with $c \geq 1$
- Intuition : secret key will be a short vector or good basis, but other reasonably short vectors or good bases can act as equivalent secret keys
- Note that NP-hardness is not known for these parameters, so we need to **assume** that these problems are hard

Worst case vs Average case hardness

- NP-hardness refers to worst-case hardness
- In cryptography we want average case hardness since we need some entropy on the keys
- Average case hard $\Rightarrow$ worst case hard, but not other way around in general
- Interesting property of lattice-based cryptography : worst-case to average-case reductions ! (see later)

Other lattice problems

- **Gap SVP** : for approximation factor $\gamma > 1$ and radius r, returns YES if $\lambda_1 \leq r$, return NO if $\lambda_1 \geq \gamma r$, and may return YES or NO otherwise
- **ISVP** : find vectors with norms equal to **successive minima** : $\lambda_k(L)$ is the smallest λ such that there are at least k linearly independent vectors with norms at most λ
- And many others...

Modular lattices

- A lattice is **modular** if $\exists q < \det(L)$ with $L \supset q\mathbb{Z}^n$
- In cryptography we often use

$$L_{A,q} = \{x \in \mathbb{R}^n | Ax = 0 \bmod q\}$$

  for some matrix $A \in \mathbb{Z}^{m\times n}$ with entries reduced modulo q
- Typically $n \approx m \log m$

(Caution : here columns of A are not lattice vectors !)

SIS

- **Small integer solution** (SIS) : given q, A and ν, find x with $Ax = 0 \bmod q$ and $||x|| \leq \nu$
- A short vector in $L_{A,q}$ gives a solution to SIS
- SIS harder when A has less columns and more rows
- SIS has solutions when ν and n large enough

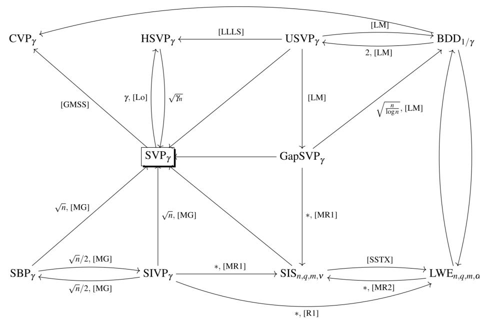
Learning with errors (LWE)

- Let q a modulus and let $s \in \mathbb{Z}_q^n$
- Let $B << q$ some noise bound
- LWE sample is (a, t) with a uniformly chosen in $\mathbb{Z}_q^n$, e uniformly chosen in $[-B, B]$, and $t = \langle a, s\rangle + e$
- **LWE problem** : given m samples (a_i, t_i), recover s
- Could use linear algebra if $B = 0$
- Other distributions for e can be used
  (in fact, we usually use Gaussian distributions)

Learning with errors (2)

- CVP-type problem for the matrix A generated by a_i :
  Given A and t, find $As \in L$ such that $e = t - As$ is small
  (in fact *bounded distance decoding* : such solution exists)
- Extension of **Learning Parity with Noise**,
  a NP-hard problem from coding theory
- **Decision LWE :** given samples (a_i, t_i) that are either LWE samples or random samples, guess distribution

Some relationships between lattice problems



Laarhoven, van de Pol, de Weger, Solving Hard Lattice Problems and the Security of Lattice-Based Cryptosystems
Arrow from Problem A to Problem B means "Problem A can be solved using an algorithm for Problem B"

Ideal lattices

- Lattice-based schemes need to include a basis of the lattice in the public key, typically n^2 coefficients
- Ideal lattices :
  - Choose a polynomial ring $R = \mathbb{Z}[x]/f(x)$ (typically $f(x) = x^n + 1$ and $n = 2^e$)
  - See a vector $v = (v_0, \ldots, v_{n-1})$ as a polynomial $v(x) = v_0 + v_1x + v_2x^2 + \ldots + v_{n-1}x^{n-1}$ in that ring
  - Ideal lattice is generated by $x^i v(x) \bmod f(x)$
  - Only store the n coefficients of v

Ideal lattices are modular

- Taking Hermite normal form, we get $q \in \mathbb{Z} \cap \langle v(x) \rangle$
- Deduce $qx^i \in \langle v(x) \rangle$ hence $L \supset q\mathbb{Z}^n$

Outline

Outline

Remember : hash functions

$$H : \{0,1\}^* \times K \rightarrow \{0,1\}^n$$

- A hash function satisfies
  - **Collision resistance**
    if hard to find m, m' such that $H_k(m) = H_k(m')$
  - **Preimage resistance**
    if given h, hard to find m such that $H_k(m) = h$
  - **Second preimage resistance**
    if given m, hard to find m' such that $H_k(m') = h$

  for a uniformly generated key $k \in K$
- We usually build a fixed-length hash function and then use Merkle-Damgaard transform

Ajtai's hash functions

- Key generation : choose a random modular lattice

$$L_{q,A} = \{x \in \mathbb{R}^n | Ax = 0 \bmod q\}$$

- Define $H : \{0,1\}^n \rightarrow \mathbb{Z}_q^m : x \rightarrow Ax \bmod q$
- Collisions $Ax = Ax'$ implies solving SIS **on average** $A(x - x') = 0 \bmod q$ with $(x - x') \in \{-1, 0, 1\}^n$ small

Worst case to average case reduction

- Goal : solve **any** instance of $\tilde{O}(n)$-SIVP given an algorithm that solves **random** instances of SIS
  (γ-SIVP = finding n linearly independent lattice vectors, the largest one being as small as possible, up to factor γ)
- Let B a lattice basis, defining an SIVP problem
- Consider parallelepiped $\mathcal{P}(B)$ consisting of all points of $\mathbb{R}^n$ modulo B
- Divide $\mathcal{P}(B)$ into q^n regularly spaced cells
- Associate cells to $\mathbb{Z}_q^n$ elements (use map $z \rightarrow f(z) = \lfloor qB^{-1}z \rceil$)

Worst case to average case reduction (2)

- Informal lemma : large enough random vectors modulo B lead to uniformly distributed points on $\mathcal{P}(B)$
  (usually take normal distributions with $\sigma = c\lambda_n$)
- Choose large enough $r_i \in \mathbb{R}^n$ with additional requirement that r_i mod B is the corner of a cell
- Provide q and $a_i = f(r_i)$ to the SIS solver and receive solution $z_i \in \{-1, 0, 1\}$ with $\sum a_i z_i = 0 \bmod q$
- Deduce lattice point $z = \sum_i r_i z_i$ with $||z||_2 \leq cn\lambda_n$
- Note that λ_n can be guessed with binary search, or take the current best approximation and repeat

Using ideal lattices

- Improve efficiency using A with special structure
- Taking circulant matrices is a bad idea
  - Lattice points correspond to elements in a principal ideal
    $$\langle a(X) \rangle \subset R = \mathbb{Z}[X]/(X^n - 1)$$
  - If $\gcd(a(X), X^n - 1) \neq 1$ then there exists $z_0 \neq 0$ with
    $$a(X)z_0(X) = 0 \bmod (X^n - 1)$$
  - Deduce collision $(z, z + z_0)$ for every z

Using ideal lattices (2)

- Solution : replace $X^n - 1$ by an irreducible polynomial
- Taking $f(X) = X^n + 1$ and $n = 2^k$ has some efficiency advantages (use Fast Fourier transform, etc)
- Security still based on worst case hardness assumptions but for **ideal** lattice problems

Further readings

- Papers by Ajtai, Lyubashevski-Micciancio, Peikert-Rosen
- Micciancio-Regev, *Lattice-based cryptography*

Outline

GGH cryptosystem : basic idea

- Private key is well-chosen good basis of a lattice (basis with short, nearly orthogonal vectors)
- Public key is well-chosen bad basis A for the same lattice (for example, the Hermite normal form of the lattice)
- Encryption of m is $As + m$, for well-chosen s (so that result is reduced modulo Hermite basis)
- Decryption is LWE / CVP like problem (in fact bounded distance decoding), easy given the private key but hard otherwise

GGH cryptosystem : remarks

- Similar to McEliece's code-based cryptosystem (1978)
- Probabilistic by padding the message with random noise (for example $m \to m + 2r$)
- No formal reduction to a hard problem and original parameters broken, but eventually led to LWE schemes
- Not CCA secure (given a ciphertext, can re-randomize it and ask the decryption oracle for plaintext)
- Can use hash functions / random oracles to transform CPA encryption into CCA encryption (Fujisaki-Okamoto)

NTRU cryptosystem (sketch)

- Let p, q coprime integers with $p << q$
- Let $R = \mathbb{Z}[X]/(X^n - 1)$
- Private key : polynomials $f, g \in R$ with small coefficients such that f invertible modulo p and q
- Public key : $h = pf^{-1}g \bmod q$
- Encryption of small $m \in R$: take random small $r \in R$ and return $c = m + hr \bmod q$
- Decryption of c is $m' = (cf \bmod q)\, f^{-1} \bmod p$
- Correctness : modulo q we have $cf = mf + pgr$ and right-hand term is small so no reduction modulo q

NTRU : link with lattices

- Public key is
$$A = \begin{pmatrix} I & 0 \\ H & qI \end{pmatrix}$$
where H is cyclic matrix corresponding to h
- Private key is short vector corresponding to f, g. Equivalently a matrix
$$B = \begin{pmatrix} F & \tilde{F} \\ G & \tilde{G} \end{pmatrix}$$
where F, G are cyclic matrices corresponding to f, g and $\tilde{F}, \tilde{G}$ are well-chosen matrices so that $\mathcal{L}(A) = \mathcal{L}(B)$
- Encryption of m is $(-r, m)^T$ modulo $\mathcal{L}(A)$

NTRU : security

- Recommended parameters (Wikipedia, citing NTRU website)

	N	q	p
Moderate Security	167	128	3
Standard Security	251	128	3
High Security	347	128	3
Highest Security	503	256	3

- No security proof for original scheme
- If secret polynomials are generated in a proper way then becomes CPA-secure under ideal lattice assumptions (see Stehlé-Steinfeld 2011)

LWE-based cryptosystem

- Parameters : integers n, m, ℓ, t, r, q and real $\alpha > 0$
- Let $f : \mathbb{Z}_t^\ell \to \mathbb{Z}_q^\ell$ defined by
$$z \to f(z) = [(q/t)z]$$
"rounded scaling" (here $q > t$)
- Let $f_{-1} : \mathbb{Z}_q^\ell \to \mathbb{Z}_t^\ell$ defined by
$$z \to f_{-1}(z) = [(t/q)z]$$
"inverse" of f

LWE-based cryptosystem (2)

- Private key is $S \in \mathbb{Z}_q^{n\times\ell}$ uniformly random
- Public key is $(A, P) \in \mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^{m\times\ell}$ with
  - $P = AS + E$
  - $E \in \mathbb{Z}_q^{m\times n}$ normal distribution with $\sigma = \alpha q/\sqrt{2\pi}$
  - $A \in \mathbb{Z}_q^{m\times n}$ uniformly random
- Encryption of $v \in \mathbb{Z}_t^\ell$ is
$$(u, c) = (A^T a, P^T a + f(v))$$
with a uniformly random in $\{-r, \ldots, r\}^m$
- Decryption of (u, c) is
$$v' = f_{-1}(c - S^T u)$$

LWE-based cryptosystem (3)

- Kind of lattice version of ElGamal
- Correctness : we have

$$\begin{aligned} c - S^T u &= P^t a + f(v) - S^T A^T a \\ &= (AS + E)^T a + f(v) - S^T A^T a) \\ &= E^T a + f(v) \end{aligned}$$

hence $f_{-1}(c - S^T u) = v$ as long as

$$||E^T a||_\infty < q/2t$$

Security

- Distinguishing (A, P) from uniformly random pairs implies solving Decisional LWE
- Encryptions with random pairs leak no information on messages (when $\#\text{inputs} = (2r+1)^m >> \#\text{outputs} = q^{n+\ell}$)
- Together these two observations imply CPA security (if you distinguish two ciphertexts then the keys are not random)
- Concrete hardness of LWE : see Albrecht-Player-Scott
- CCA encryption scheme follows from generic reductions such as Fujisaki-Okamoto (more direct constructions now exist)

Further readings

- Micciancio-Regev, *Lattice-based cryptography*
- Peikert, *A decade of lattice cryptography*

Outline

Digital signatures : basic idea

- Private key is a good basis B of a lattice
- Public key is a bad basis for the same lattice
- Let H a collision resistant hash function with image in $\mathbb{R}^n$
- To sign, compute $H(m)$, use nearest plane algorithm (see later) with good basis to obtain close lattice point s, and return it
- To verify, check that s and $H(m)$ are close
- Examples : GGH signatures, NTRU signatures

Digital signatures : improvements

- Basic idea broken [Nguyen-Regev]
  - Signature (m, s) leaks $s - H(m)$ a uniformly distributed point in (a translation of) the fundamental parallelipiped

  - Attacker obtains several (m_i, s_i) then recovers B by solving an optimization problem
- Solution : signature a quite close vector (distance $\approx c\lambda_n$), making sure distribution of $s - H(m)$ is independent of B

Further readings

- Peikert, *A decade of lattice cryptography* and references therein

Outline

Fully homomorphic encryption (FHE)

- RSA is multiplicatively homomorphic :
  $\text{Enc}(m_1 m_2) = (m_1 m_2)^e \bmod n = \text{Enc}(m_1)\text{Enc}(m_2)$
- Additively homomorphic schemes have also been known for a long time $\text{Enc}(m_1 + m_2) = \text{Enc}(m_1) + \text{Enc}(m_2)$
- Satisfying both properties simultaneously allows cool stuff, such as statistics on encrypted data
- FHE was long-standing open problem until 2009
  First solution by Gentry, followed by many other ones
- All solutions based on lattices !

FHE key ideas

- Encrypt your messages as noisy ring elements like in previous encryption schemes based on ideal lattices
- This gives **somewhat fully homomorphic encryption**
  Homomorphic additions and multiplications of ciphertexts, but not too many as each operation increases the noise
  (hence at some point you cannot decrypt correctly anymore)
- Could decrease the noise by decrypting and re-encrypting, but that would reveal intermediary plaintexts
- **Bootstrapping :** encrypt noisy ciphertext again using somewhat homomorphic scheme, do internal decryption and re-encryption homomorphically using an encrypted decryption key, and remove second level of encryption

Simple example

- Symmetric version
  - Secret key is a large prime p
  - To encrypt a bit m, choose random $r << p$ and large q, then return $c = m + 2r + pq$
  - To decrypt c, compute $m' = (c \bmod p) \bmod 2$
  - Homomorphic $+$ and $\times$ as long as noise $<< p$
  - CPA secure if approximate gcd problem is hard
    (given several samples $pq_i + s_i$, return p)
    (can be reformulated as lattice problem)
- Asymmetric version
  - Public key has several encryptions of 0 $(c_i = 2r_i + pq)$
  - Encryption of m is $c = m + \sum_{i \in I} c_i + 2r$ for a subset I

Further readings

- Peikert, *A decade of lattice cryptography*
  and references therein

Outline

Lattices and lattice hard problems

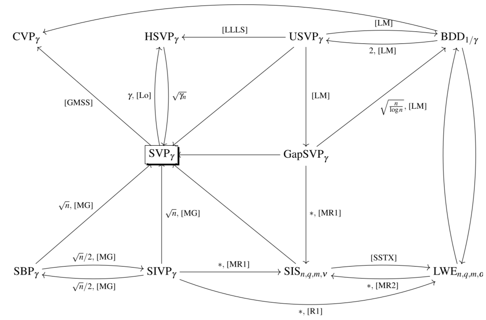
Lattice-based constructions

Solving hard lattice problems
- Lattice reduction algorithms
- Exact solvers
- Further algorithms

Hardness results on main lattice problems

Cryptanalysis applications

Some relationships between lattice problems



Laarhoven, van de Pol, de Weger, Solving Hard Lattice Problems and the Security of Lattice-Based Cryptosystems
Arrow from Problem A to Problem B means "Problem A can be solved using an algorithm for Problem B"

Outline

Lattices and lattice hard problems

Lattice-based constructions

Solving hard lattice problems
- Lattice reduction algorithms
- Exact solvers
- Further algorithms

Hardness results on main lattice problems

Cryptanalysis applications

Lenstra-Lenstra-Lovacz

- *Factoring polynomials with rational coefficients*, 1982
- The paper defines a notion of reduced basis and gives an algorithm to compute them
- Solve γ-SVP with approximation factor $2^{(n-1)/2}$

Orthogonal projections

- Given $u = (u_1, \ldots, u_n)$, $v = (v_1, \ldots, v_n) \in \mathbb{R}^n$, their scalar product is $\langle u, v \rangle = \sum_{i=1}^n u_i v_i$
- The orthogonal projection of u on v is $u_v := \frac{\langle u,v \rangle}{\langle v,v \rangle} v$
- The orthogonalization of u wrt v is

$$u_{\perp v} := u - u_v = u - \frac{\langle u, v \rangle}{\langle v, v \rangle} v$$

  We have $\langle u_{\perp v}, v \rangle = \langle u, v \rangle - \frac{\langle u,v \rangle}{\langle v,v \rangle} \langle v, v \rangle = 0$
- Define $\text{Perp}(u, \{v_1, \ldots, v_k\}) = u - \sum_{i=1}^k \frac{\langle u,v_i \rangle}{\langle v_i,v_i \rangle} v_i$
  We have $\langle \text{Perp}(u, \{v_1, \ldots, v_k\}), v_i \rangle = 0$

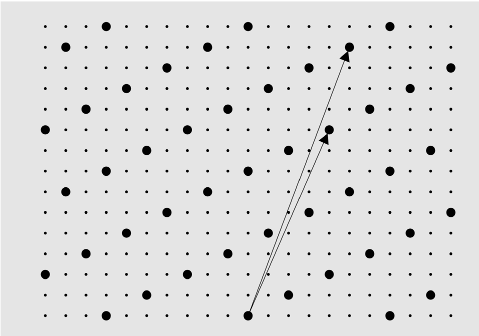
Gram-Schmidt orthogonalization

- Given a basis B, compute an orthogonal basis B^* and upper triangular matrix M with ones on the diagonal (in particular $\det M = 1$) such that $B^* = BM$
- The orthogonal basis is computed as
  $b_1^* = b_1$, $b_2^* = \text{Perp}(b_2, \{b_1^*\})$,
  $b_3^* = \text{Perp}(b_3, \{b_1^*, b_2^*\})$, $b_4^* = \text{Perp}(b_4, \{b_1^*, b_2^*, b_3^*\})$,
  etc
- For any $i > j$, we have $M_{i,j} = -\frac{\langle b_i, b_j^* \rangle}{\langle b_j^*, b_j^* \rangle}$
  (in general M will not be integer)
- May depend on the ordering of the basis vectors

LLL for $n = 2$: Gauss algorithm

- Goal : given a lattice basis $\{b_1, b_2\}$, find v in the lattice with minimal norm
- Ideas :
  - Swapping two vectors preserves the lattice
  - Adding an integer number of times one vector to the other one preserves the lattice
  - When two vectors are "nearly parallel", reducing the largest one by the smallest one provides a smaller vector

Gauss algorithm : example



Picture credit : Antoine Joux, Algebraic Cryptanalysis

Gauss algorithm

Swap b_1 and b_2 if needed to ensure $||b_1|| \geq ||b_2||$
while $||b_1|| > ||b_2||$ **do**
  $\lambda \leftarrow \lfloor \langle b_1, b_2 \rangle / \langle b_2, b_2 \rangle \rceil$
  $b_1 \leftarrow b_1 - \lambda b_2$
  Swap b_1 and b_2
end while
return (b_1, b_2)

- Similar to Euclide algorithm, continued fractions,...

Gauss algorithm : analysis

- The lattice is preserved at all steps
- The algorithm terminates
- At each step λ minimizes the value of $||b_1 - \lambda b_2||^2 = \lambda^2 \langle b_2, b_2 \rangle - 2\lambda \langle b_1, b_2 \rangle + \langle b_1, b_1 \rangle$
- Final basis (b_1, b_2) satisfies $\left| \frac{\langle b_1, b_2 \rangle}{\langle b_1, b_1 \rangle} \right| \leq \frac{1}{2}$
- Final b_1 has minimal norm
  (see Joux for details of the proof)

Reduced basis

- In dimension 2, we can say a basis is reduced when

$$||b_1|| \leq ||b_2|| \text{ and } \left| \frac{\langle b_1, b_2 \rangle}{\langle b_1, b_1 \rangle} \right| \leq \frac{1}{2}$$

  This guarantees that b_1 has minimal norm
- In larger dimension there is no similar condition (and corresponding algorithm) that guarantees that
- However, the vectors of an **LLL-reduced basis** are never too far from optimal

LLL-reduced basis

- Let $1/4 < \delta \leq 1$
- We say a basis $\{b_1, \ldots, b_n\}$ is δ-**LLL-reduced** iff

$$\forall i < j \ : \ |\langle b_j, b_i^* \rangle| \leq \frac{||b_i^*||^2}{2}$$

$$\forall i \ : \ \delta ||b_i^*||^2 \leq \left(||b_{i+1}^*||^2 + \frac{\langle b_{i+1}, b_i^* \rangle^2}{||b_i^*||^2} \right)$$

- Here b_i^* are the Gram-Schmidt basis vectors
- First condition identical to dimension 2
- Second condition is called Lovász condition

Properties of LLL basis

- The two conditions imply
$$||b_{i+1}^*||^2 \geq ||b_i^*||^2 \left(\delta - \frac{1}{4}\right)$$
- λ_1 must be at least as large as some $||b_i^*||$
- Hence for some i we have
$$\lambda_1 \geq ||b_i^*|| \geq \left(\delta - \frac{1}{4}\right)^{(i-1)/2} ||b_1^*||$$
- Hence for $\delta = 3/4$ and some i we have
$$||b_1|| = ||b_1^*|| \leq 2^{(i-1)/2}\lambda_1 \leq 2^{(n-1)/2}\lambda_1$$

Properties of LLL basis

- We have $\det(L) = \prod_i ||b_i^*||$ hence
$$\det(L) \geq \left(\delta - \frac{1}{4}\right)^{n(n-1)/4} ||b_1^*||^n$$
hence for $\delta = 3/4$
$$||b_1|| \leq 2^{(n-1)/4} \det(L)^{1/n}$$
- Similar bounds can be derived for the other b_i

LLL algorithm

- Maintains a counter k such that the basis is LLL-reduced up to index $k-1$
- Updates the basis via two operations
  - Reduction of b_k by all b_j with $j < k$ to satisfy the first condition
  - Swap of b_k and b_{k-1} if Lovacz condition not satisfied
- Maintains a Gram-Schmidt basis B^* and corresponding matrix M with respect to the current basis B
(in fact, only M and the norms of b_i are needed)

Length reduction

- Length reduction of b_i

1: **for** $j = i-1$ to 1 **do**
2: $\quad b_i \leftarrow b_i - \left\lfloor \frac{\langle b_i, b_j^* \rangle}{\langle b_j^*, b_j^* \rangle} \right\rceil b_j$
3: **end for**

- Sort of approximation of
$$\text{Perp}(b_i, \{b_1, \ldots, b_{i-1}\}) = b_i - \sum_{j=1}^{i-1} \frac{\langle b_i, b_j^* \rangle}{\langle b_j^*, b_j^* \rangle} b_j^*$$

LLL algorithm

```
Let k ← 2
while k ≤ n do
b_k ← LengthReduce(b_k, {b_1, ..., b_{k-1}})
if Lovacz condition holds for i = k − 1 then
    k ← k + 1
else
    Swap b_{k-1} and b_k
    k ← max{2, k − 1}
end if
end while
return (b_1, ..., b_n)
```


Complexity (sketch)

- Let d_i be the determinant of the ith sublattice generated by basis vectors $b_1, \ldots, b_i$
- $d_i = \prod_{j=1}^{i} ||b_j^*||^2$
- Consider the quantity $D = \prod_{i=1}^{n} d_i$
- D only changes when there is a swap
- At each swap of b_k and b_{k-1},
  $||b_{k-1}^*||^2$ is decreased by a factor at least δ^{-1},
  d_{k-1} is decreased by a factor at least δ^{-1},
  and none of the other d_i changes
- D cannot be arbitrary small, so LLL must stop

Improvement : BKZ

- Stronger notion of reduced basis : Korkine Zolotarev, giving the shortest vector
- Corresponding algorithm has exponential time
- Block Korkine Zolotarev : variant of LLL with exact SVP search on sublattices $\langle b_k, b_{k+1}, \ldots, b_{k+r}\rangle$
- Lead to shorter vectors at some efficiency cost
- Requires efficient *exact* solvers in larger dimensions !
- See CP Schnorr, *Block Korkin-Zolotarev Bases and Successive Minima*

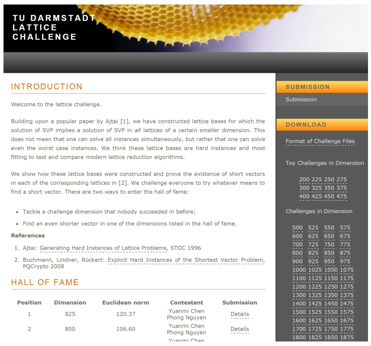
A folklore statement

- Lattice-reduction algorithms perform much better in practice than what is predicted by the theory

Gama-Nguyen experiments

- Goal was to evaluate folklore statement
- Warning : experiments necessarily on certain lattice distributions, basis distributions, limited size parameters
- Some observations :
  - Approximation factor of LLL and other algorithms is γ^n, exponential in dimension as predicted by theory, but with a much lower constant γ than predicted
  - In practice γ is small enough that $\gamma^n \approx 1 + (\gamma - 1)n$ when $n < 450$, and n-SVP could be solved for those lattices

Lattice reduction hall of fame



Lattice-reduction hall of fame

- Methodology to generate lattice challenges
- Challenges solved by research teams around the world, competing to appear in the "Hall of Fame"
- Goal is to find the shortest possible vectors in lattice challenges
- Also adapted to ideal lattices
- See `http://www.latticechallenge.org/`

Outline

Exact solvers

- Exact solvers not directly needed as approximate solutions usually enough to break lattice-based schemes
- However, approximate solvers also use exact solvers on smaller problems internally
- Two main approaches for exact SVP
  - Enumeration
  - Sieving
- Note that exact solvers can also be accelerated with an approximate solver pre-processing step

Principle of enumeration

- Identify a finite set of possible solutions
- Perform (intelligent) brute force on it

Enumeration bounds

- Let b_i be a basis for L and let b_i^* the corresponding Gram-Schmidt basis
- We search for $\alpha_i \in \mathbb{Z}$ such that $v = \sum_{i=1}^{n} \alpha_i b_i$ has minimal norm
- Given any $v' \in L$, we know $||v|| \leq ||v'||$
- $v = \sum_{i=1}^{n} \beta_i b_i^*$ for some n, where $\beta_n = \alpha_n$ and $\beta_i \in \mathbb{R}$
- From $||v||^2 = \sum_{i=1}^{n-1} \beta_i^2 ||b_i^*||^2 + \alpha_n^2 ||b_n^*||^2 \leq ||v'||^2$, we deduce $|\alpha_n| \leq \frac{||v'||}{||b_n^*||}$
- So only a finite number of options to test for α_n !

Enumeration bounds

- For each α_n possible value, we can iterate the reasoning and find a bound on $|\alpha_{n-1}|$, etc
- Only a finite number of options to test for all α_i !
- Note that as we find smaller and smaller vectors we also decrease our search space

Preprocessing with lattice reduction

- Starting from an LLL-reduced basis is a good idea :
  - Taking $v' = b_1$ leads to a small $||v'||$
  - The last b_i^* are the largest ones
  - Hence $|\alpha_k| \leq \frac{||v'||}{||b_k^*||}$ are smaller
- So better to do LLL or BKZ before enumerating !

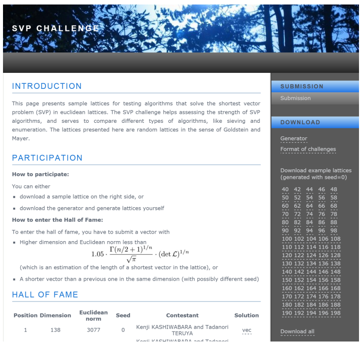
Pruning

- Idea : remove some branches of the enumeration tree with a certain probability when they are "unlikely" to contain a shortest vector
- For example, it is unlikely that all components are as large as the bounds allow
- Can miss the shortest vector with some probability
- Extreme pruning by Gama-Nguyen : compensate for low probabilities by repeating the search

Sieving

- Idea of sieving : maintain a long list of reasonably short vectors in the lattice, and combine them pairwise to obtain some even shorter vectors
- Lead to exponential running time algorithms (vs super-exponential running time for enumeration) but they also require exponential space
- See D. Micciancio and P. Voulgaris, *A Deterministic Single Exponential Time Algorithm for Most Lattice Problems based on Voronoi Cell Computations*
- Or *Solving Hard Lattice Problems and the Security of Lattice-Based Cryptosystems* for a short description

SVP Hall of Fame



SVP Hall of Fame

- Note that SVP are not known by the challenge organizers, so Gaussian heuristic approximation is used to assess the quality of short vectors
- Also adapted to ideal lattices
- See `http://www.latticechallenge.org/`

Outline

Combinatorial solvers

- Suppose we want to find vectors with coordinates bounded by b in the modular lattice

$$L_{A,q} = \{x \in \mathbb{R}^n | Ax = 0 \bmod q\}$$

defined by the matrix $A \in \mathbb{Z}^{m \times n}$

- Can use Wagner's generalized birthday algorithms
- Sometimes more efficient than lattice reduction

Generalized Birthday Attacks

- Divide A into 2^k groups of $n/2^k$ columns
- For each group, build a list with all linear combinations with coefficients in $\{-b, \ldots, b\}$
- There are $L = (2b+1)^{n/2^k}$ vectors per list
- Combine the lists pairwise as follows
  - Take all sums $v_1 + v_2$ with v_i in list i
  - Keep sums where first $\log_q L$ coordinates are 0
- Keep about L elements on average, since there are L^2 sums and L values for first coordinates

Generalized Birthday Attacks (2)

- We now have 2^{k-1} lists with roughly L elements
- Combine them again and again, until you get one list of vectors that are 0 in the $k \log_q L$ coordinates
- One element in the last list is expected to have $(k+1)\log_q L$ coordinates at 0
- To solve SIS problem choose k such that

$$m \approx (k+1)\log_q L$$

Babai's nearest plane algorithm

- Goal is to solve γ-approximate closest vector problem : given B, t, find $x \in \mathcal{L}(B)$ close to t
- Use LLL then reduce t by lattice vectors

1: $B \leftarrow \text{LLL}(B)$
2: $b \leftarrow t$
3: **for** $j = n$ **to** 1 **do**
4: $\quad b \leftarrow b - \left\lfloor \frac{\langle b, b_j^* \rangle}{\langle b_j^*, b_j^* \rangle} \right\rceil b_j$
5: **end for**
6: **return** $x = t - b$

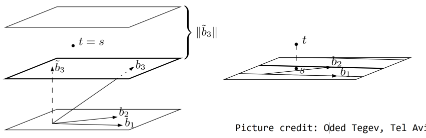
- Achieves approximation $\gamma = 2(2/\sqrt{3})^n$

Babai's nearest plane algorithm (2)

- Nearest plane algorithm : after initial LLL step
  - Find $\lambda = \left\lfloor \frac{\langle b, b_n^* \rangle}{\langle b_n^*, b_n^* \rangle} \right\rceil$ such that hyperplane

$$\lambda b_n^* + \text{span}(b_1, \ldots, b_{n-1})$$

  is as close as possible to b
  - Recurse on $b - \lambda b_n$ and $\mathcal{L}(b_1, \ldots, b_{n-1})$



Picture credit: Oded Tegev, Tel Aviv course 2004

Analysis (sketch)

- Goal : prove that $||x - t|| \le 2^{n/2} d(t, B)$
- Let $y \in \mathcal{L}$ a closest lattice vector
- Goal is to prove $||x - t|| \le 2^{n/2} ||y - t||$
- Proof by recursion on the dimension
  - When $n = 1$ closest vector is returned
  - Larger n : either λ is "correct guess" or not, namely either $y \in \lambda b_n + \text{span}(b_1, \ldots, b_{n-1})$ or not

Case $y \in \lambda b_n + \text{span}(b_1, \ldots, b_{n-1})$

- Let $t' =$ projection of $(t - \lambda b_n)$ on $\text{span}(b_1, \ldots, b_{n-1})$
- Babai on $(t', \{b_1, \ldots, b_{n-1}\})$ returns $x' = x - \lambda b_n$
- Since $y \in \lambda b_n + \text{span}(b_1, \ldots, b_{n-1})$ then $y' := y - \lambda b_n$ is closest vector to t' in sublattice
- By induction we have

$$||x' - t'|| \le 2^{(n-1)/2}||y' - t'||$$

- We deduce

$$\begin{aligned} ||x - t||^2 &= ||x' - t'||^2 + ||t - \lambda b_n - t'||^2 \\ &\le 2^{n-1}||y' - t'||^2 + ||t - \lambda b_n - t'||^2 \\ &\le 2^n \left(||y' - t'||^2 + ||t - \lambda b_n - t'||^2\right) \\ &= 2^n||y - t||^2 \end{aligned}$$

Case $y \notin \lambda b_n + \text{span}(b_1, \ldots, b_{n-1})$

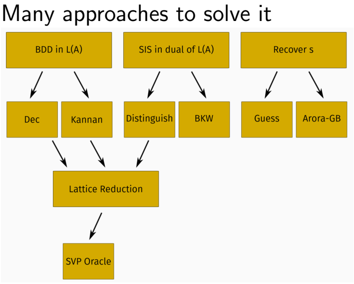
- Let $d_k = ||t - (kb_n + \text{span}(b_1, \ldots, b_{n-1}))||$
- $d_k \le \frac{1}{2}||b_n^*||$ when $k = \lambda$, and $d_k > \frac{1}{2}||b_n^*||$ when $k \neq \lambda$
- So $||y - t|| > \frac{1}{2}||b_n^*||$
- By construction we have $||x - t||^2 \le \frac{1}{4}\sum_{i=1}^n ||b_i^*||^2$
- From LLL basis properties with $\delta = 3/4$

$$||x - t|| \le \frac{1}{2}2^{n/2}||b_n^*||$$

- We deduce $||x - t|| \le 2^{n/2}||y - t||$
- Can improve γ by changing LLL parameters

LWE solvers

- Many approaches to solve it



Credit picture: Martin Albrecht, UCL 05/07/15 talk

- Concrete hardness still an open problem !

Outline

Lattices and lattice hard problems

Lattice-based constructions

Solving hard lattice problems

Hardness results on main lattice problems

Cryptanalysis applications

Are SVP and CVP hard ?

- Decisional CVP is NP-hard
- Search and Decisional CVP are equivalent
- Search and Decisional SVP are equivalent
- Can solve SVP if can solve CVP
- Heuristically the converse if also true

Solving Search CVP with Decisional CVP

- Lemma : Search CVP can be solved in polynomial time given an oracle that solves Decisional CVP
- Let B and t be a search CVP instance
- First recover $r = d(t, \mathcal{L}(B))$
  - Notice $r \leq R = \sum_i ||b_i||$ and $r^2 \in \mathbb{Z}$
  - Use binary search and Decision SVP oracle to find r
- Then recover $v \in \mathcal{L}(B)$ such that $||v - t|| = r$
  (a) Find $t' = t - u$ with $u \in \mathcal{L}(B)$ and $d(t', 2^k B) = r$ with $k = n + \log r$
  (b) Find $w \in \mathcal{L}(2^k B)$ with $||w - t'|| = r$
  (c) Return $v = u + w$

Solving (a) : iterative procedure

- Goal : find $t' = t - u$ with $u \in \mathcal{L}(B)$ and $d(t', 2^k B) = r$ with $k = n + \log r$
- Given $B = \{b_1, b_2, \dots, b_n\}$ build $B' = \{2b_1, b_2, \dots, b_n\}$
- Call Decisional CVP oracle on B', t and r
  - If $d(\mathcal{L}(B'), t) = r$ then keep t as it is
  - If $d(\mathcal{L}(B'), t) \neq r$ then $d(b_1 + \mathcal{L}(B'), t) = r$, in other words $d(\mathcal{L}(B'), t - b_1) = r$, so replace t by $t - b_1$
- Repeat this procedure, building sparser and sparser lattices, and t' as required

Solving (b) : nearest plane algorithm

- Goal : find $w \in \mathcal{L}(2^k B)$ with $||w - t'|| = r$
- This w exists by construction
- Distance between any two vectors in $\mathcal{L}(2^k B)$ at least $2^n \cdot r$
- Second closest vector at distance at least

$$2^n \cdot r - r \geq 2^{n-1} \cdot r$$

- Apply nearest plane algorithm to get a closest vector up to approximation bound smaller than 2^{n-1}, hence the closest vector
- Polynomial time reduction

Decisional CVP is NP-complete

- Decisional CVP is in NP : witness is closest lattice point, solution checked in polynomial time
- Decisional CVP is NP-hard : reduction from the subset sum problem

Subset-sum problem

- Subset-sum problem : given integers a_i and target sum S, find a subset of the a_i that sum up to S
- Often called knapsack problem in cryptography
- Equivalent decision variant : decide if there is a solution
- Equivalent to have $S = 0$
- Equivalent to consider sums modulo an integer
- Problem NP-hard in general

Decisional Subset Sum from Decisional CVP

- Let $a_1, \ldots, a_n, S$ defining a decisional subset sum problem
- Build the decision CVP instance defined by

$$B = \begin{pmatrix} a_1 & a_2 & \ldots & a_n \\ 2 & 0 & \ldots & 0 \\ 0 & 2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \ldots & 0 & 2 \end{pmatrix} \qquad t = \begin{pmatrix} S \\ 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} \qquad r = \sqrt{n}$$

- Return answer from decisional CVP instance

Analysis

- Consider lattice vectors Bx with $x_i \in \{0, 1\}$
- If we have $\sum_i a_i x_i = S$ then
  - First coordinate of $Bx - t$ is 0
  - Other coordinates are ± 1
  - $||Bx - t|| \le \sqrt{n}$
  - Decisional CVP oracle returns yes
- If decisional CVP oracle returns yes then
  - There is x with $||Bx - t|| \le \sqrt{n}$
  - First coordinate of $Bx - t$ is 0 and other ones are ± 1
  - We have $\sum_i a_i x_i = S$

Decisional SVP from Decisional CVP

- Let B, r defining a decisional SVP instance
- Suppose we can solve Decisional CVP instances
- Let B_i generated by $(b_1, \ldots, b_{i-1}, 2b_i, b_{i+1}, \ldots, b_n)$
- Use Decisional CVP oracle on B_i, b_i, r for all i
- Return YES iff DCVP oracle returns YES at least once

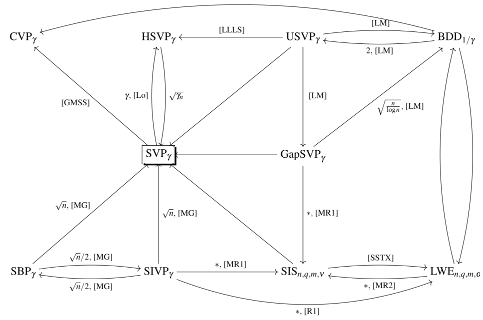
Analysis

- Assume $\lambda_1(\mathcal{L}(B)) > r$
  - Let $i \in \{1, \ldots, n\}$ and $v \in \mathcal{L}(B_i)$
  - We have $v - b_i \in \mathcal{L}(B)$ and $v - b_i \neq 0$
  - By assumption $||v - b_i|| > r$
  - Hence oracle returns NO for all i
- Assume $\lambda_1(\mathcal{L}(B)) \leq r$
  - Let shortest vector $v = \sum_i a_i b_i$ with $a_i \in \mathbb{Z}$ and $||v|| \leq r$
  - At least one a_i is odd, otherwise v not shortest
  - Let k such that a_k is odd
  - Then $b_k + v \in \mathcal{L}(B_k)$
  - Then $d(b_k, \mathcal{L}(B_k)) \leq ||v|| = r$
  - Hence oracle returns YES for $i = k$

Computational CVP from Computational SVP

- Let B, t be a computational CVP instance
- Expand all basis vectors by a 0 coordinate
- Expand target vector by a 1 coordinate
- Solve Computational SVP problem for a basis containing all expanded vectors including the target one
- Heuristically, we expect a short vector in the new lattice to be short in its first components
- Remark : SVP problem slightly bigger dimension

Some relationships between lattice problems



Laarhoven, van de Pol, de Weger, Solving Hard Lattice Problems and the Security of Lattice-Based Cryptosystems
Arrow from Problem A to Problem B means "Problem A can be solved using an algorithm for Problem B"

Further readings

- Micciancio-Goldwasser, *Complexity of lattice problems*
- Oded Regev's lecture notes at Tel Aviv university, 2004

Outline

Outline

Subset-sum problem

- Subset-sum problem : given integers a_i and target sum S, find a subset of the a_i that sum up to S
- Often called knapsack problem in cryptography
- Equivalent decision variant : decide if there is a solution
- Equivalent to have $S = 0$
- Equivalent to consider sums modulo an integer
- Problem NP-hard in general

Merkle-Hellman cryptosystem

- Private key is an easy knapsack instance a_i, and two integers r and q
- Example of easy knapsack : superincreasing sequence $a_i > \sum_{j<i} a_j$
- Public key is knapsack instance $b_i = a_i r \bmod q$
- Message bits define a subset ; encryption is subset sum
- Decryption of c : solve easy knapsack for $c/r \bmod q$

Knapsack cryptosystems and lattices

- Knapsack cryptosystems were broken with lattices
- On the other hand, knapsack cryptosystems can also be seen as ancestors of current lattice-based cryptosystems

Short relations

- Goal : given vectors v_i, find small λ_i such that $\sum \lambda_i v_i = 0$
- Build a lattice generated by the columns of matrix

$$\begin{pmatrix} Kv_1 & Kv_2 & \dots & Kv_r \\ 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & 1 \end{pmatrix}$$

- Lattice elements are $(K \sum \lambda_i v_i; \lambda_1; \dots; \lambda_r)$
- If K is large enough, the first components of small vectors must be 0

Analysis

- The lattice contains vectors with 0 first components, and other vectors
- Expected size of shortest vector can be bounded, say $||\lambda|| < B$, using pigeonhole principle
- LLL will find a vector v in the lattice with length smaller than $B2^{(n-1)/2}$
- Any vector in the lattice with nonzero first component has length at least K
- Choose $K > B2^{(n-1)/2}$ such that LLL will necessarily return a vector with 0 first components

Knapsack hash function

- Fix some integers a_i
- $H : \{0,1\}^n \to \mathbb{Z} : x \to \sum_i x_i a_i$
- Can break H by finding collisions, that are messages (x, x') with $\sum_i x_i a_i = \sum_i x'_i a_i$
- Attack : build the lattice as before (with $v_i = a_i$), and hope to get a small vector $(0, \lambda_1, \ldots, \lambda_r)$ with $\lambda_i \in \{-1, 0, 1\}$
- Attack only heuristic but parameters with 128 numbers of 120 bits each can be broken in practice [Joux]

Short modular relations

- Goal : given vectors v_i and N, find small λ_i such that $\sum \lambda_i v_i = 0 \bmod N$
- Build a lattice

$$\begin{pmatrix} Kv_1 & Kv_2 & \ldots & Kv_r & KNI \\ 1 & 0 & \ldots & 0 & 0 \\ 0 & 1 & \ldots & 0 & 0 \\ 0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & \ldots & 1 & 0 \end{pmatrix}$$

where I is an identity matrix

- Lattice elements are $(K(\sum \lambda_i v_i + N \sum \mu_i e_i), \lambda_1, \ldots, \lambda_r)$
- If K is large enough, the first component of small vectors must be 0

Outline

RSA with small decryption key

- Using a small decryption key d for RSA is appealing for efficiency reasons, moreover
  - If d has 80 bits then exhaustive search not possible
  - If $n = pq$ is large enough then factoring is not possible
- Is this secure ?

Small root attacks

- Problem : given a polynomial f modulo an integer N **with small roots**, compute these roots
- The small root condition is crucial : no hope to compute roots of $x^2 - 1$ in general, equivalent to factoring N
- Application to RSA with small decryption key

$$de = k\varphi(N) + 1 = k(N - z) + 1$$

where $z = O(\sqrt{N})$ and d, k are "small"

Don Coppersmith

Don Coppersmith

From Wikipedia, the free encyclopedia

Don Coppersmith (born c. 1950) is a cryptographer and mathematician. He was involved in the design of the Data Encryption Standard block cipher at IBM, particularly the design of the S-boxes, strengthening them against differential cryptanalysis.[1] He has also worked on algorithms for computing discrete logarithms, the cryptanalysis of RSA, methods for rapid matrix multiplication (see Coppersmith–Winograd algorithm) and IBM's MARS cipher. Don is also a co-designer of the SEAL and Scream ciphers.

Also invented small root attacks. . .

Small root attacks : idea

- Let us start with f univariate
- Solving polynomials modulo N is hard, but solving them over $\mathbb{Z}$ is easy
- $f(x) = 0 \bmod N \;\Rightarrow\; g(x)f(x) = 0 \bmod N$ for all g
- If $h(x) = 0 \bmod N$ and $\left|\sum_i h_i x^i\right| \leq \sum_i |h_i| \cdot |x|^i \leq N$ then $h(x) = 0$ over the integers
- Idea : find $h = gf$ with small values of $|h_i| \cdot |x|^i$ using LLL on a well-chosen lattice

Building a lattice

- Let B be a bound on $|r|$
- In fact, we will consider equations satisfied modulo powers of N instead of just N to facilitate lifting to $\mathbb{Z}$
- Let $F_{i,j,k}(x) = x^i f(x)^j N^k$
- If $f(r) = 0 \bmod N$ then $F_{i,j,k}(r) = 0 \bmod N^{j+k}$ and the same is true for their linear combinations
- Let $D, t \in \mathbb{N}$ to be fixed later
- Let $\mathcal{F} := \{F_{i,j,t-j} \mid \deg F_{i,j,t-j} \leq D\}$
- We have $F(r) = 0 \bmod N^t$ for all $F \in \mathcal{F}$

Building a lattice

- To any F with $\deg F \leq D$, we associate a vector
$$v_F = (F_0, F_1 B, F_2 B^2, \ldots, F_D B^D)'$$
- Let L be the lattice generated by $\{v_F \mid F \in \mathcal{F}\}$
- Any vector $v \in L$ is equal to v_F for some F such that
$$F = \sum_{i,j} a_{i,j} F_{i,j,t-j}$$
- This F satisfies $F(r) = 0 \bmod N^t$

Short vectors

- A short vector in L corresponds to F such that
$$||v_F||_2 = ||(F_0, F_1 B, F_2 B^2, \ldots, F_D B^D)'||_2$$
is small
- This also implies $||v_F||_1 = \sum_{i=0}^{D} |F_i| B^i$ is small
- If $||v_F||_1 \leq N^t$ then $F(r) = 0$ over the integers
- If $F(r) = 0$ over the integers, we can compute its roots, including the roots of f

Analysis (sketch)

- Take $D = (t+1)\deg f - 1$
- Evaluate the determinant of L as
$$\det(L) = N^{(D+1)t/2} B^{D(D+1)/2}$$
- LLL can return v satisfying
$$||v||_2 \leq 2^{D/4} N^{t/2} B^{D/2}$$
- Translate this bound to L1 norm
- Deduce it works as long as $(B\sqrt{2})^D \approx N^t$
- For large t we can achieve $B \approx N^{1/\deg f}/\sqrt{2}$

Bivariate polynomials

- Let $f(x,y)$ and bounds B_x and B_y
- Now construct polynomials $F_{i,j,k,\ell} = f(x,y)^i x^j y^k N^\ell$
- Construct a lattice in a same way
- Recover the two smallest vectors instead of just one
- Deduce a system of two equations in x and y over $\mathbb{Z}$
- Heuristically, can recover x, y by solving this system
- Analysis is more complex, but except for the last step everything is guaranteed to work

RSA with small decryption key

- Using a small decryption key for RSA is appealing for efficiency reasons, however we have

$$de = k\varphi(N) + 1 = k(N - z) + 1$$

where $z = O(\sqrt{N})$ and d, k are small
- Wiener's attack using continued fractions
- Improvements by Boneh-Durfee using lattices

Factoring with implicit hints

- Suppose you know some continuous bits of p and/or q
- Suppose two RSA moduli share some continuous bits
- Can reduce these problems to some polynomial equation with small roots

Further references

- Joux, *Algorithmic cryptanalysis*, Chapter 13.2, and references therein

Outline

DSA, ECDSA and ElGamal signatures

- Parameters : cyclic group G of order p, generator $g \in G$, and a mapping $f : G \to \mathbb{Z}_p$
- Secret key is random $x \in \mathbb{Z}_p$
- Public key is $h = g^x$
- Signature of message $m \in \mathbb{Z}_p$
  - Pick random $y \in \mathbb{Z}_p$
  - Find b such that $m = by - xf(g^y) \bmod p$
  - Return (m, g^y, b)
- Verification : check that

$$g^{mb^{-1}} h^{f(g^y)b^{-1}} = g^y$$

Attack model

- Attacker receives several signatures (m_i, g^{y_i}, b_i)
- Attacker also receives some bits of each y_i, for example they know

$$y_i = z_i' + 2^\lambda z_i + 2^\mu z_i''$$

entirely except for z_i with $0 \le z_i < B = 2^{\mu-\lambda}$

Small root problem

- Deduce several equations in z_i and x

$$m_i = b_i(z_i' + 2^\lambda z_i + 2^\mu z_i'') - xf(g^{y_i}) \bmod p$$

- Eliminate x to get equations

$$z_i = s_i z_0 + t_i \bmod p$$

for some known s_i, t_i
- Obtain a system of equations, with solutions smaller than expected from random systems of this size

Lattice reduction step

- Build a lattice generated by columns of

$$A = \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ s_1 & p & 0 & & \vdots \\ s_2 & 0 & p & & \vdots \\ \vdots & \vdots & & \ddots & \vdots \\ s_n & 0 & 0 & \dots & p \end{pmatrix}$$

- Use nearest plane algorithm to find a vector close to

$$t = (0, t_1, t_2, \dots, t_n)'$$

Analysis (sketch)

- By construction there is a lattice vector Au with

$$Au - t = (z_0, z_1, \ldots, z_n)'$$

hence $||Au - t|| \leq \sqrt{(n+1)}B$

- Nearest plane algorithm returns lattice vector w with

$$||w - t|| \leq c_1 ||b^*_{n+1}||$$

(we proved $c_1 \leq 2^{(n-1)/2}$)

Analysis (sketch)

- Except for small vector Au we heuristically expect the lattice to follow Gaussian heuristic, hence

$$||b^*_{n+1}|| \approx c_2 \det(A)^{1/(n+1)} = c_2 p^{n/(n+1)}$$

for some small $c_2 > 1$

- If $\sqrt{(n+1)}B < c_1 c_2 p^{n/(n+1)}$ then
  - Au is within range of nearest plane algorithm
  - We don't expect any other vector to be that close

Remarks

- Neglecting factors $\sqrt{(n+1)}$ and $c_1 c_2$ we get condition

$$B < p^{n/(n+1)}$$

- Attack works if we know a fraction $\epsilon = 1/(n+1)$ of y_i
- Time complexity better for smaller n
- Attack can be generalized to different bit patterns
- Bits of y_i can be obtained from side-channel attacks, weak pseudorandom generators,...

Outline

Conclusion on Lattice-based cryptography

- Lattice problems are appealing
  - Worst case to average case reductions
  - Mostly resist quantum computers so far
  - Basic problems are NP-hard
- Lattice problems are useful
  - Signature and encryption schemes, hash functions
  - Fully homomorphic encryption, multilinear maps
- Original motivation was cryptanalysis
- Very active research field, now moving towards practice

Open problems

- Faster, smaller, simpler, more secure constructions
- Classical and quantum resistance
  - Note that parameters used in cryptography are **not** believed to be NP-hard
  - Practical parameter evaluation is underway
  - We now have a quantum algorithm for special lattices
- Many DPhil challenges !