Cryptography from worst-case complexity assumptions

Daniele Micciancio
UC San Diego

LLL+25
June 2007 (Caen, France)

Outline

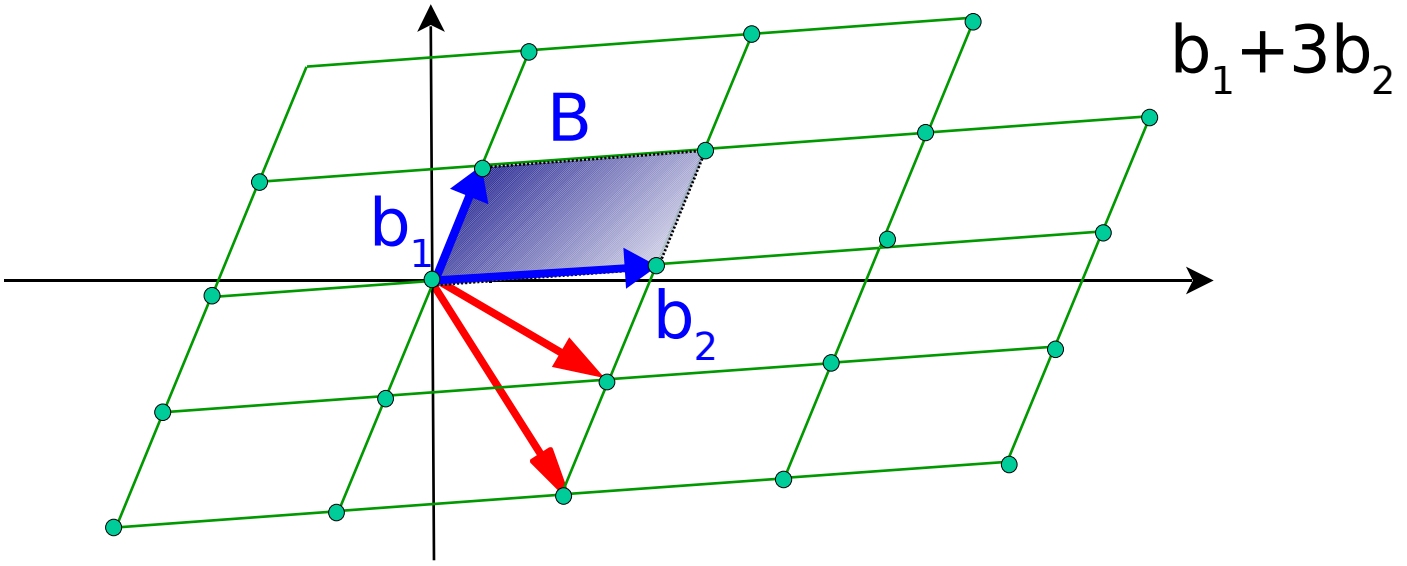
- Introduction
  - Lattices and algorithms
  - Complexity and Cryptography
- Lattice based cryptography
  - Lattice based hash functions
  - Other cryptographic primitives
- Conclusion / Open Problems
  - Choosing security parameters
  - Using lattices with special properties

Outline

- Introduction
  - Lattices and algorithms
  - Complexity and Cryptography
- Lattice based cryptography
  - Lattice based hash functions
  - Other cryptographic primitives
- Conclusion / Open Problems
  - Choosing security parameters
  - Using lattices with special properties

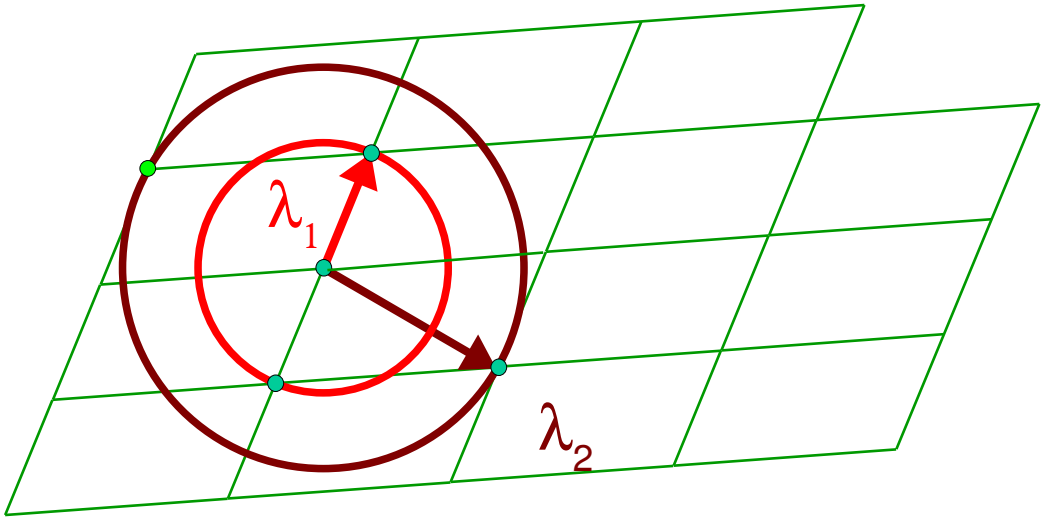
Point Lattices

- Set of all integer linear combinations of basis vectors $B = [b_1,\ldots,b_n] \subset R^n$
- $L(B)=\{Bx: x \in Z^n\} \subset \text{span}(B)=\{Bx: x \in R^n\}$



Successive Minima

- For every n-dimensional lattice L, and *i=1,…,n*, the i^{th} successive minimum $\lambda_i(L)$ is the smallest radius r such that Ball(0,r) contains *i* linearly independent lattice vectors



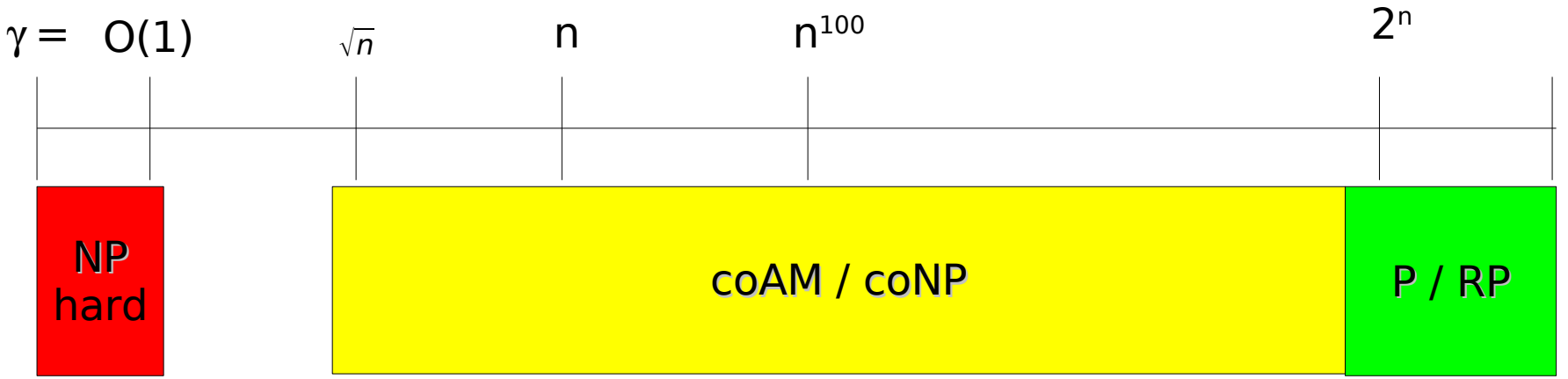
Lattice problems

- Shortest Vector Problems (SVP)
  - Given a lattice L, find the nonzero lattice vector v closest to the origin ($||v|| \leq \gamma \lambda_1(L)$)
- Shortest Independent Vect. Prob. (SIVP)
  - Given a lattice L, find n lin. independent vectors $v_1,\ldots,v_n$ of length $\max_i ||v_i|| \leq \gamma \lambda_n(L)$
- Approximation factor $\gamma(n)$ usually a function of the lattice dimension n.

Lattice Reduction Algorithms

- [LLL] solves SVP_γ and $SIVP_\gamma$ for $\gamma = 2^{O(n)}$
  - Still useful in many algorithmic applications
- [Sch,NS] Improve polynomial running time of LLL
- [Sch,AKS] Improve $\gamma = 2^{O(n \log(n) / \log \log (n))}$
- This talk
  - Assume no efficient algorithm solves lattice problems substantially better than LLL
  - Application: design cryptographic functions

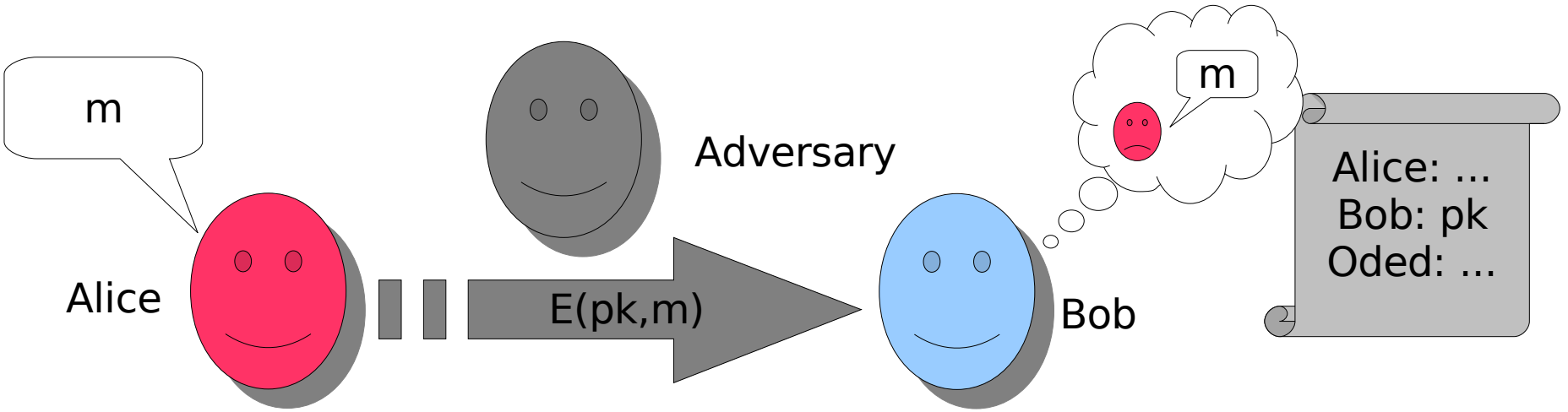
Complexity of SVP, SIVP, CVP



- NP-hard [vEB, Ajt, ABSS, Mic, BS, K]
- coAM, coNP [GG, AR, GMR]
- P, RP [LLL, Sch, AKS]
- Conjecture: SVP, SIVP are hard for $\gamma = n^{O(1)}$
  - not NP-hard, but still hard (e.g., not in P)

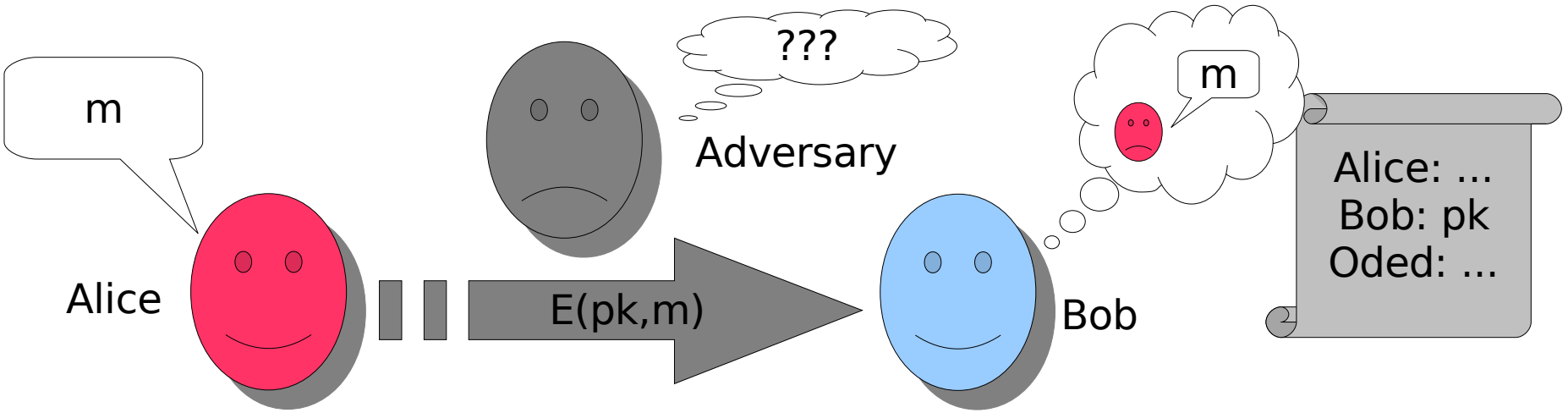
Cryptography by examples (1)
Public Key Encryption

- Alice wants to send m to Bob, *privately*
  - Bob generates (pk,sk), and publishes pk
  - Alice retrieves pk, and send E(pk,m) to Bob
  - Bob uses sk to retrieve m = D(sk,E(pk,m))



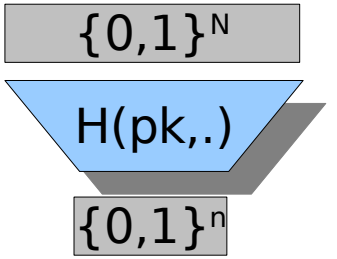
Cryptography by examples (1) Public Key Encryption

- Alice wants to send m to Bob, *privately*
  - Bob generates (pk,sk), and publishes pk
- Security:
  - Computing m from pk & E(pk,m) is hard with high probability, when pk is *randomly* chosen

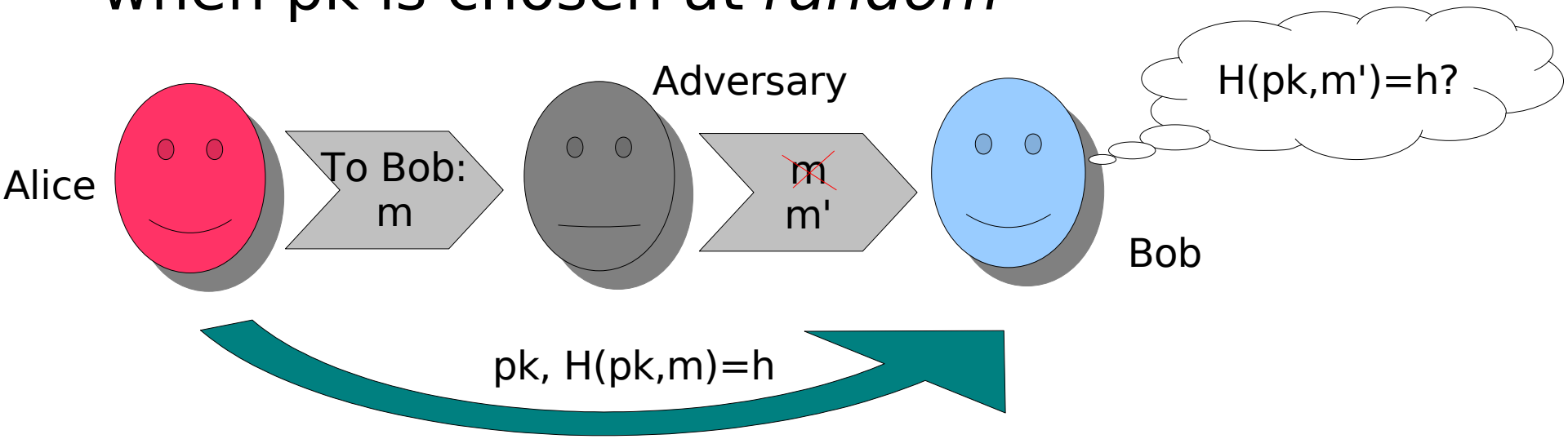


Cryptography by examples (2) Collision Resistant Hashing

- H(pk,m): No sk! Only pk.
  - H(pk, $\{0,1\}^N$) = $\{0,1\}^n$, N>>n



- Security:
  - finding collisions H(pk,m) = H(pk,m') is hard when pk is chosen at *random*



“Provable security” approach to Cryptography

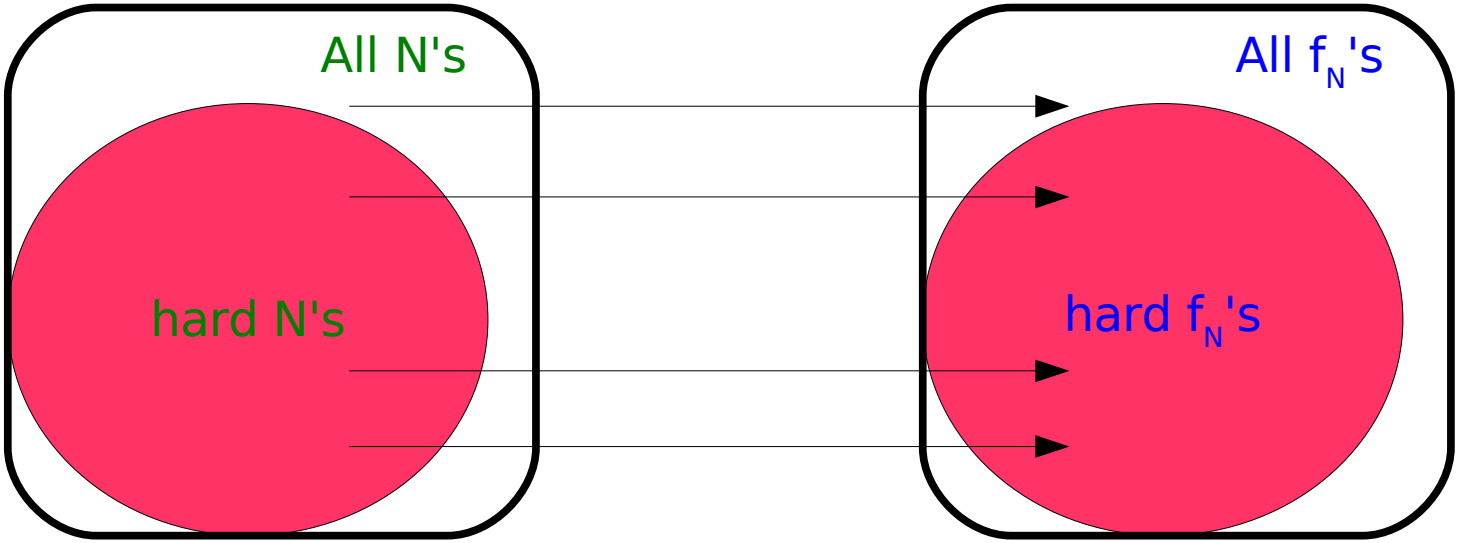
- Start from a hard computational problem
  - e.g., factoring large prime product $N=pq$
- Define a cryptographic function that is somehow related to the hard problem
  - e.g., modular squaring $f(x) = x^2 \bmod N$
- Reduce solving the hard problem to breaking the cryptographic function
  - If you were able to compute square roots mod N, then you could efficiently factor N

Worst-case vs. Average-case

- Worst-case complexity
  - A problem can be solved in time T(n) if there is an algorithm with running time T(n) that solves all problems instances of size n
  - Used in algorithms and complexity: P,NP,etc.
- Average-case complexity
  - There is an algorithm that solves a large fraction of the instances of size n
  - Used in cryptography: assume there is no such algorithm

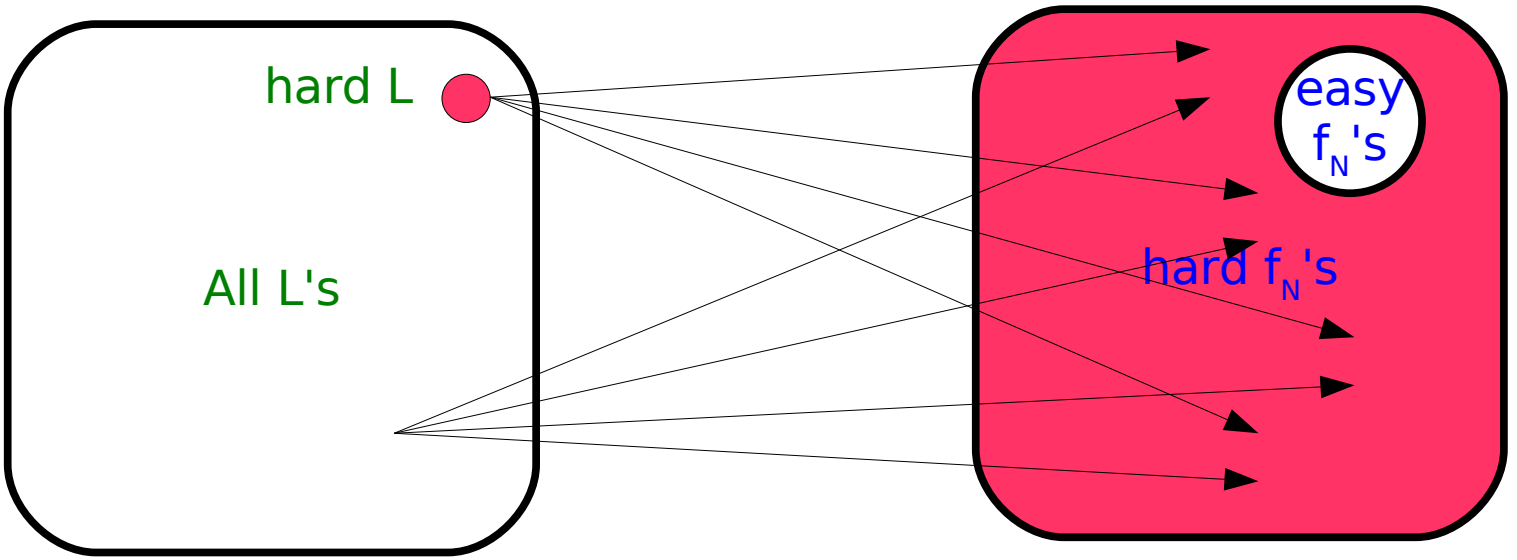
Provable security from average case hardness

- Example: (Rabin) modular squaring
  - $f_N(x) = x^2 \bmod N$, where $N=pq$, ...
  - Inverting f_N is as hard as factoring N
- f_N is cryptographically hard to invert, provided *most* N are hard to factor



Provable security from worst case hardness

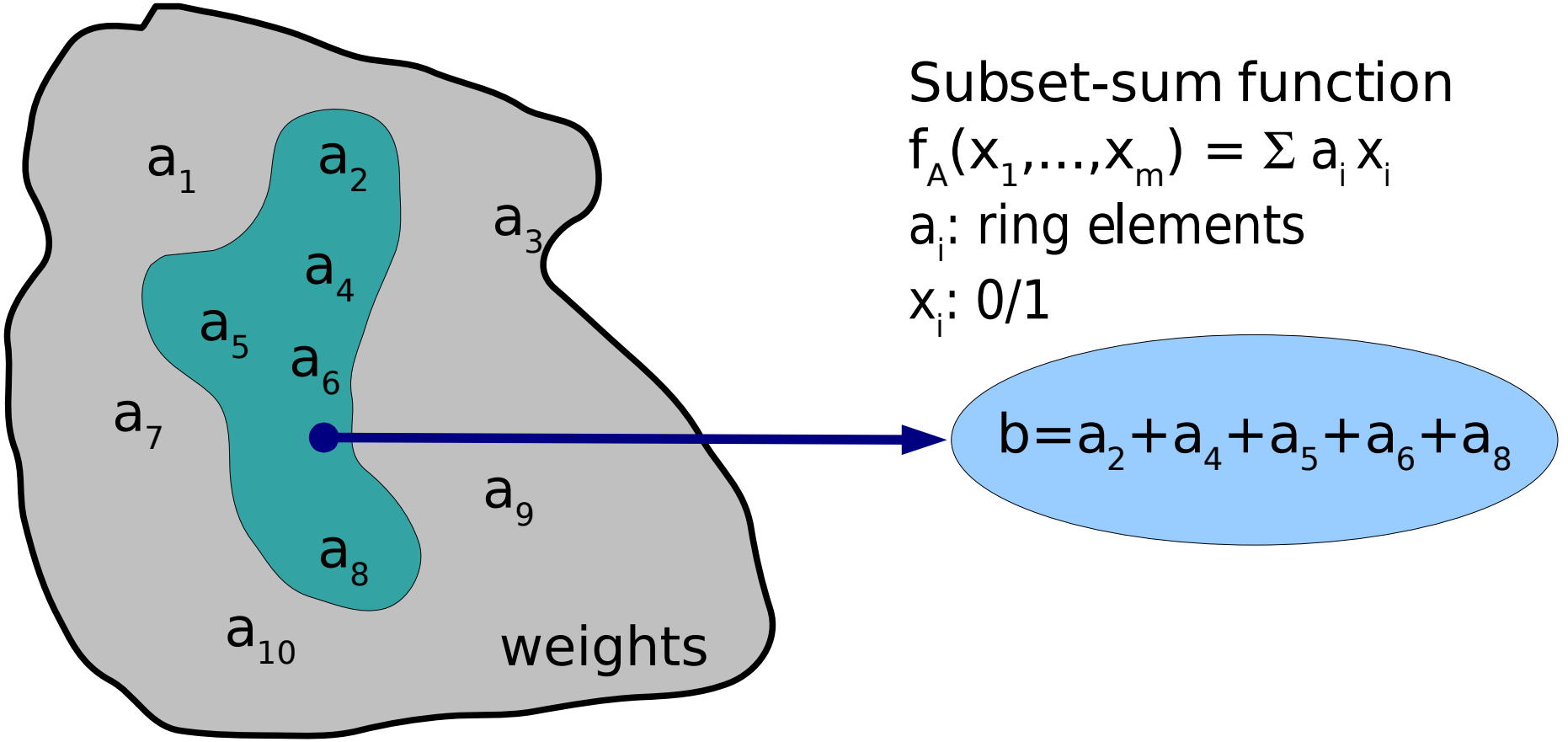
- Any fixed L is mapped to random f_N
- f_N is cryptographically hard to invert if lattice problem L is hard in the worst case



Outline

- Introduction
  - Lattices and algorithms
  - Complexity and Cryptography
- Lattice based cryptography
  - Lattice based hash functions
  - Other cryptographic primitives
- Conclusion / Open Problems
  - Using lattices with special properties
  - Choosing security parameters

The Subset-sum Problem

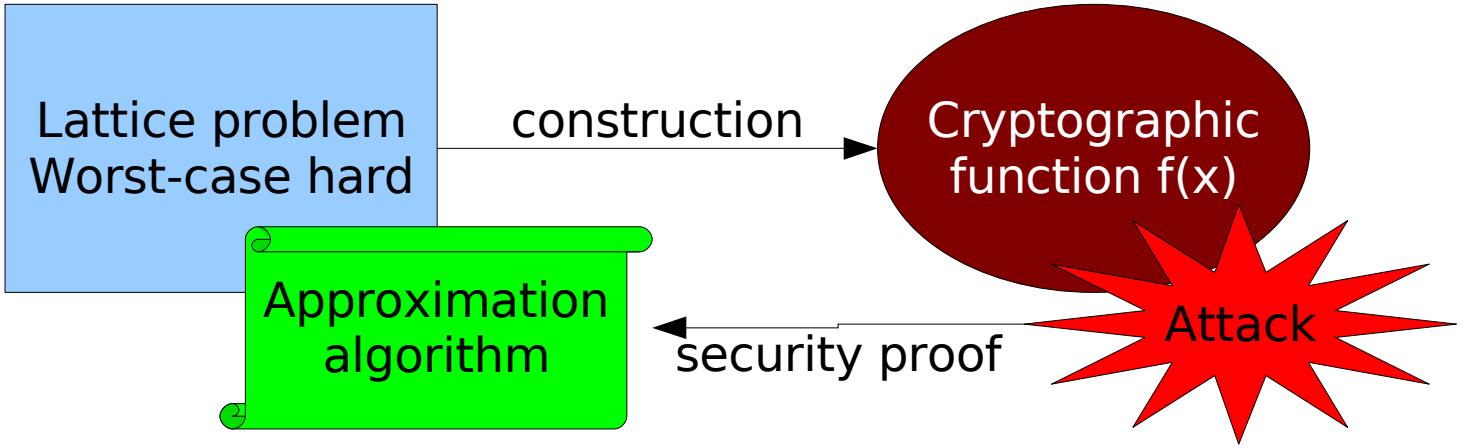


Subset Sum Problem: Given weights $A = (a_1,\ldots,a_m)$ and target b, find coefficients $x_1,\ldots,x_m$ such that $f_A(x_1,\ldots,x_m) = b$.

Subset-sum Hash function

- Key: $A = [a_1,...a_m]$ where a_i is in group G
- Input: $x=(x_1,...,x_m)$ where $x_i=0/1$
- Output $f_A(x) = Ax = \Sigma a_i x_i$
- Collisions:
  - 0/1 vectors x, y such that $Ax = Ay$
  - Equiv.: -1/0/1 vector $z =(x-y)$ such that $Az=0$
- Parameters:
  - $m > \log_2 |G|$, e.g., $G = Z^n/(pZ^n)$, $m =2n \log(p)$

Lattice based cryptography

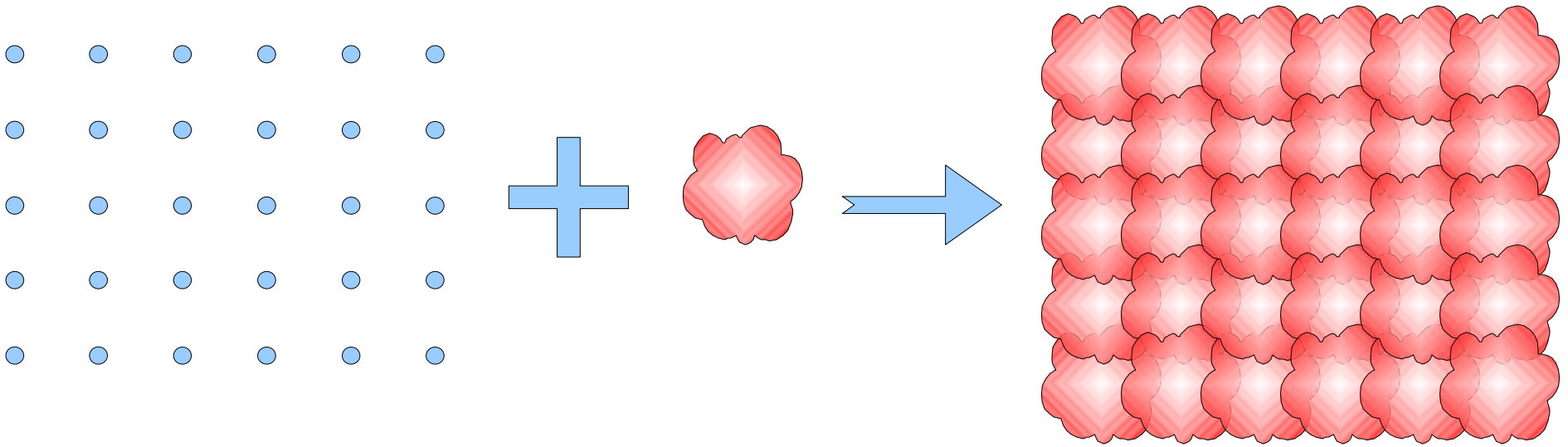


- Proof of security:
  - Assume can break random function $f_A(x)$
  - Use attack(A) to solve SIVP on any lattice B
- Main problem
  - A need to depend on B
  - A should be uniformly random, given B

Lattice based Hash function (oversimplified version)

- Construction:
  - Subset-sum over R
  - Key: random points $a_1,\ldots,a_m$ in R^n
  - Function: $f_A(x_1,\ldots,x_m) = \Sigma_i\, a_i x_i$, $(x_i$ in $\{0,1\})$
  - $f_A : \{0,1\}^m \dashrightarrow R^n$
- Technical problem
  - Range R^n is infinite, so f_A never compresses
  - We will address this using $Z_M^{\,n}$ instead of R^n

Intuition



LATTICE

random noise

$\mathbf{R^n}$

Every point in R^n can be written as the sum

$a = v + r$

of a lattice point v and small error vector r

Security proof

- Proof of security:
  - Generate random key as $a_i = v_i + r_i$ (i=1,...n)
  - Find a collision $f_A(x_1,...,x_m) = f_A(y_1,...,y_m)$
  - Notice: $\Sigma_i\, a_i\, z_i = 0$, where $z_i = x_i - y_i$
- Substituting $a_i = v_i + r_i$ and rearranging:

$$\Sigma_i\, v_i\, z_i = -\Sigma_i\, r_i\, z_i$$

Lattice vector

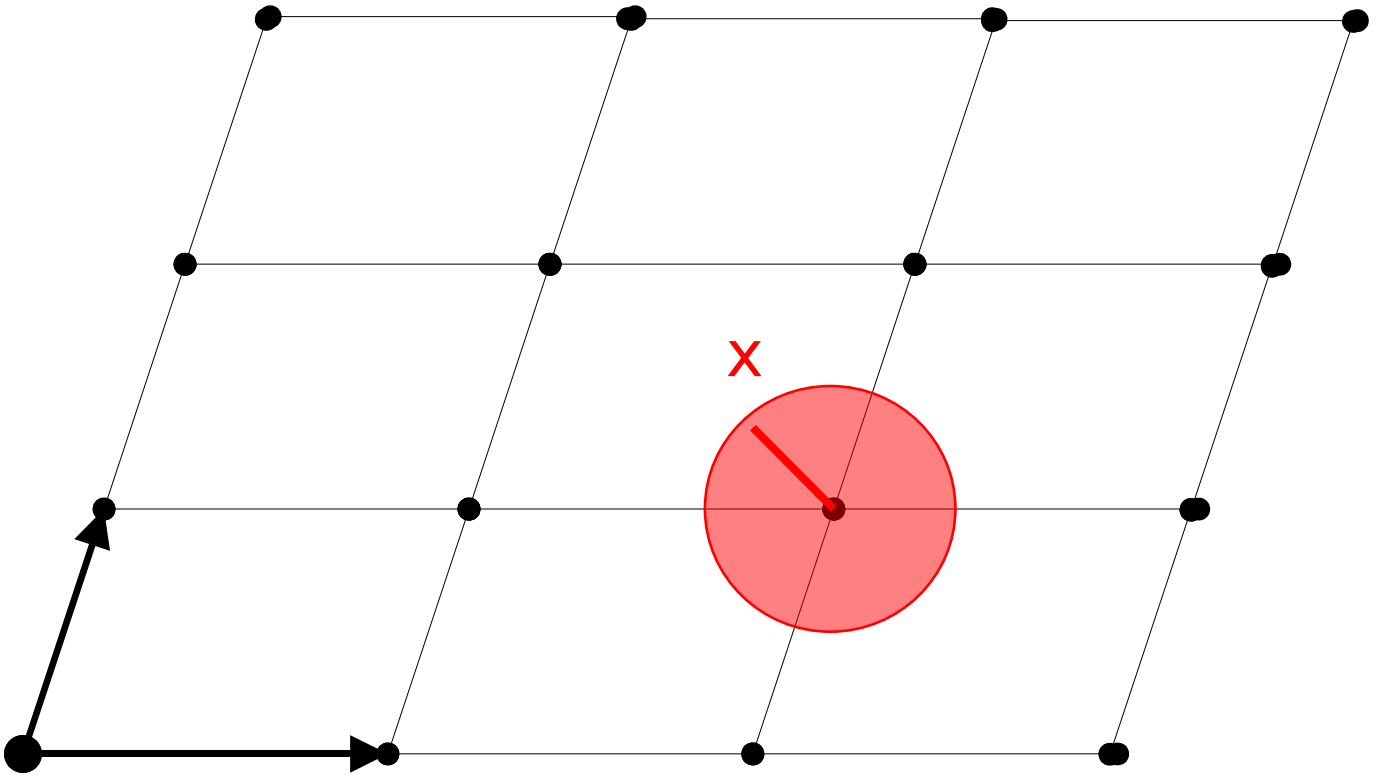
short vector

Technical details

- Issues with oversimplified construction
  - Cannot pick uniformly random lattice point v_i
  - Range of the function R^n is infinite
- Solution
  - Work "modulo L(B)"
  - Use fine grid Z^n/q instead of R^n
- Final result
  - $f_A(x) = Ax \bmod q$, where A is in Z_q^{mxn}

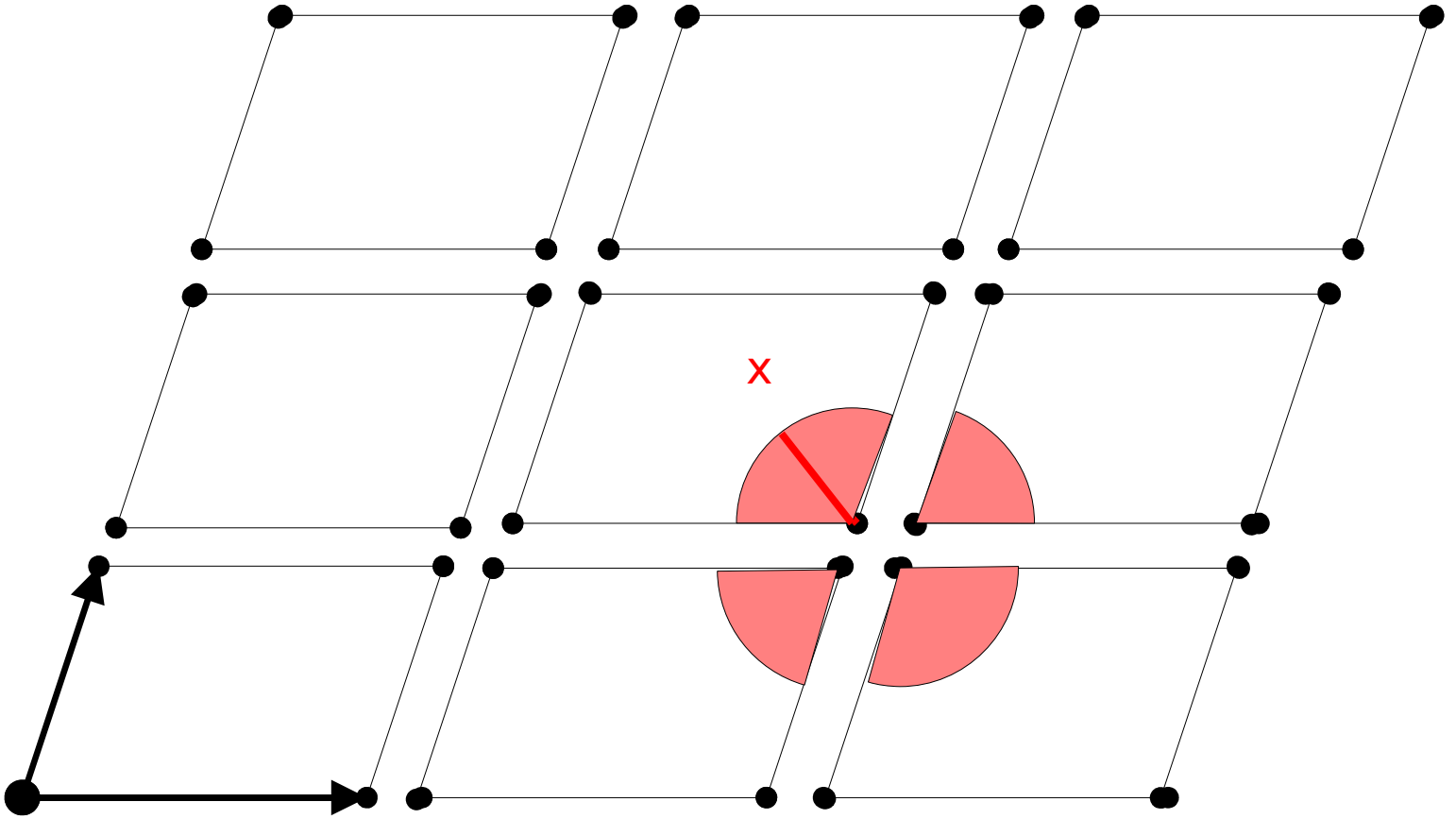
Adding noise to “all” lattice points

- Reducing a point modulo a lattice



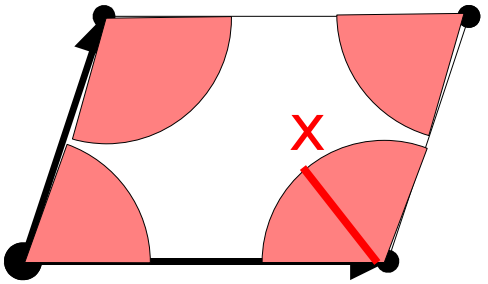
Adding noise to “all” lattice points

- Reducing a point modulo a lattice



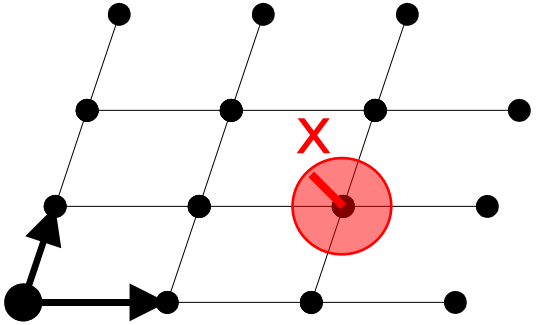
Adding noise to “all” lattice points

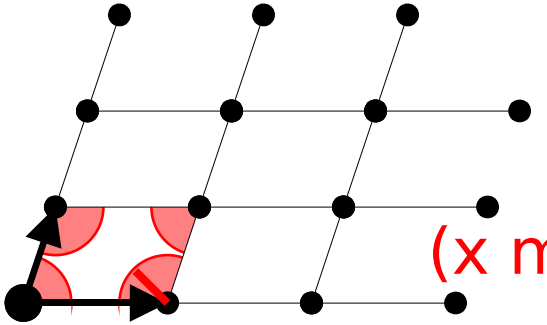
- Reducing a point modulo a lattice



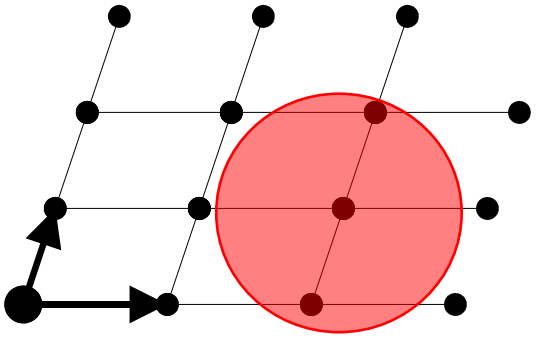
Error vector modulo the lattice

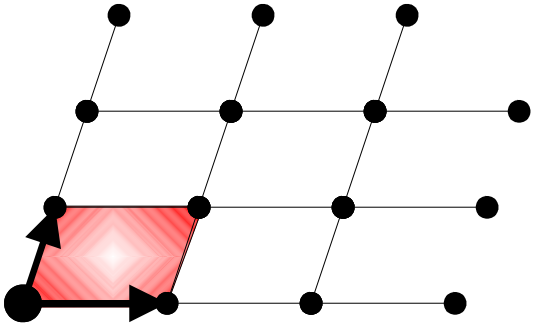
- Same as adding noise to all lattice points



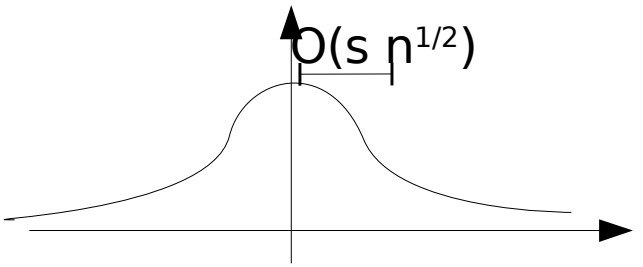


- How much noise is needed to get almost uniform distribution modulo B?





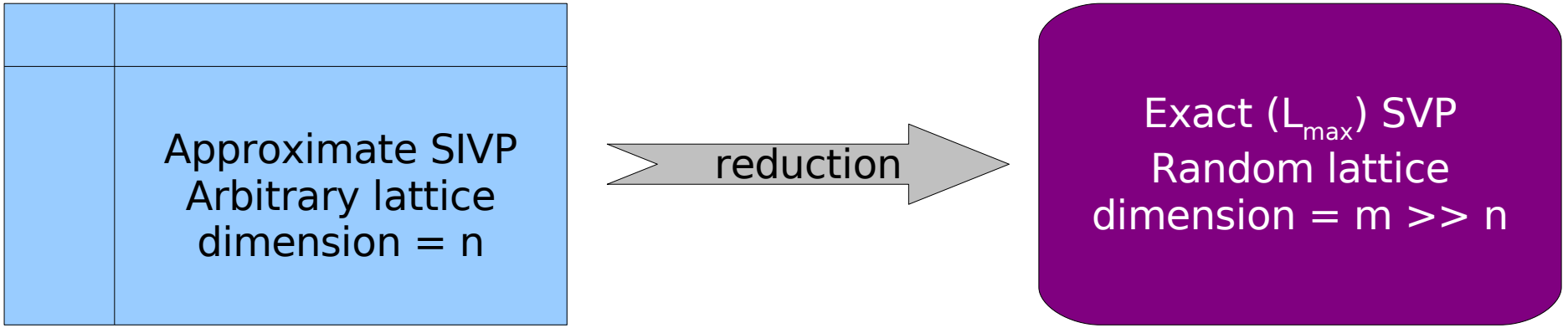
Smoothing parameter [MR]



- Gaussian $\rho_s(x) = \exp(-\pi \|x/s\|^2)$
- Smoothing parameter
  - $\eta_\epsilon(L(B))$ = smallest s s.t. $\rho_{1/s}(L(B)^* \setminus \{0\}) \leq \epsilon$.
- Properties:
  - For $s = \eta_\epsilon(L(B))$, distribution ($s^{-n}\rho_s$ mod B) is within $\varepsilon/2$ distance from uniform over P(B)
  - $\eta_\epsilon(L(B)) < \log(n)\, \lambda_n(L(B))$

Remark: Worst to Average case connection

- The set $L = \{z \text{ in } Z^m \mid f_A(z)=0\}$ is a lattice
- Collisions: z=x-y in L of norm $||z||_{max} = 1$
- Security proof:



Worst-case complexity assumption

Average-case cryptanalysis

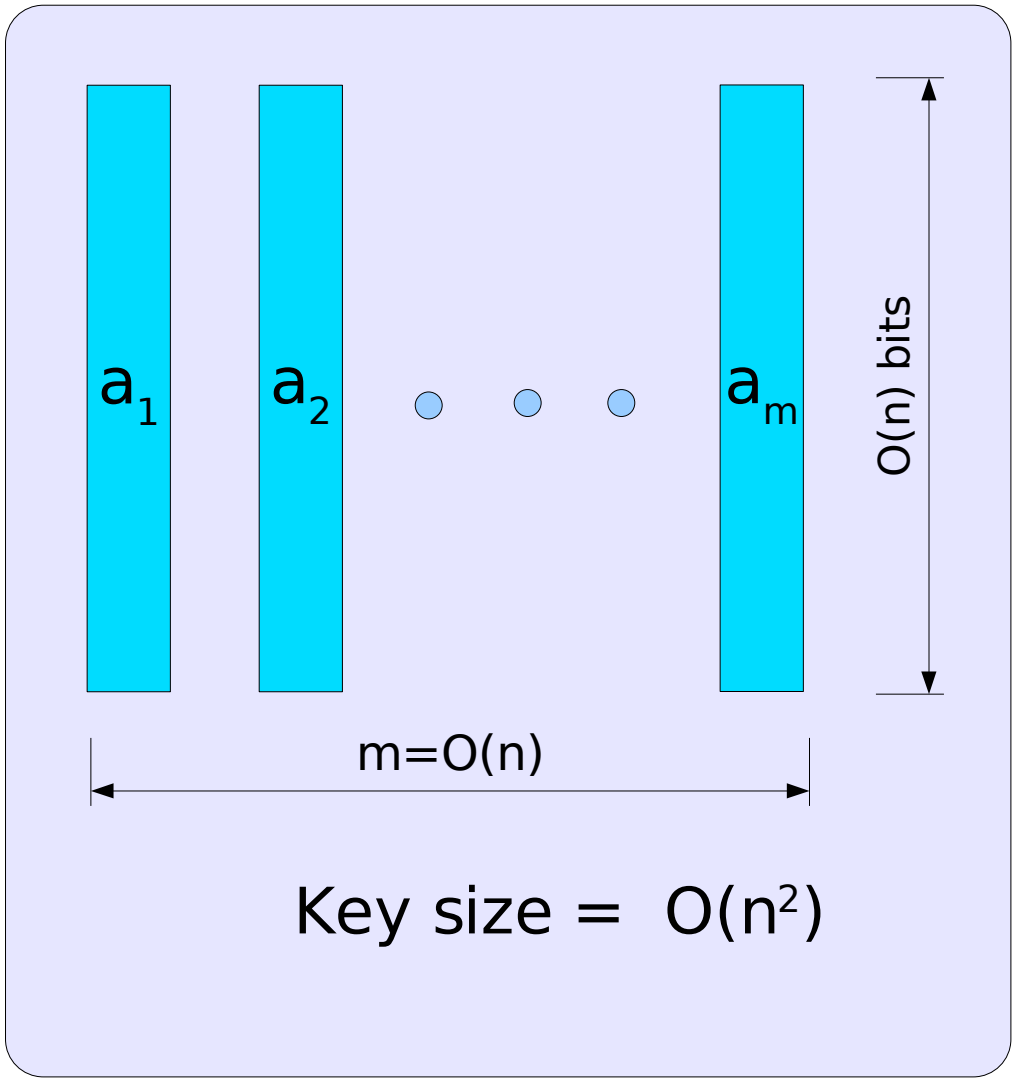
More Crypto from Lattices

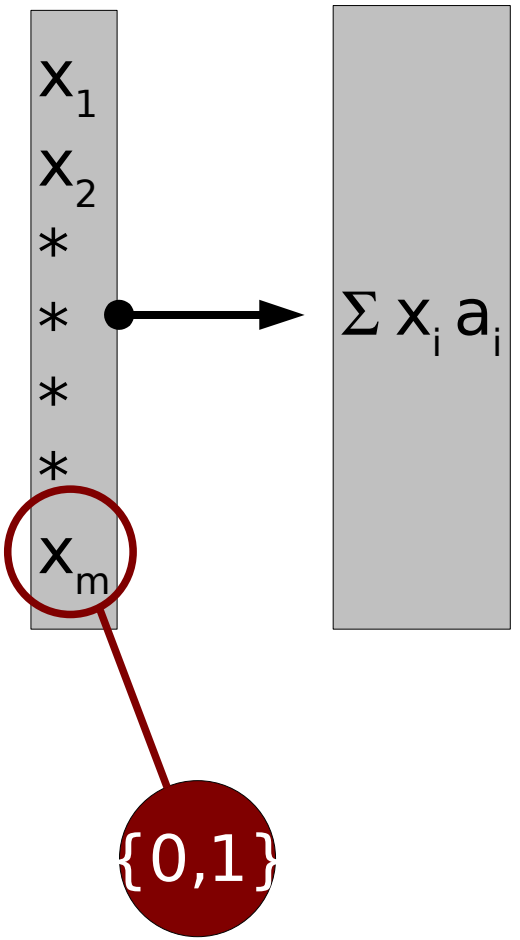
- One-way hash functions
  - [Ajt], [CN], [Mic], [MR]
- Public key encryption
  - [AD, Reg] public key encryption based on hardness of uSVP
- More efficient constructions
  - [Mic], [PR], [LM] hash functions with almost linear complexity based on hardness of cyclic lattices

Public Key Encryption

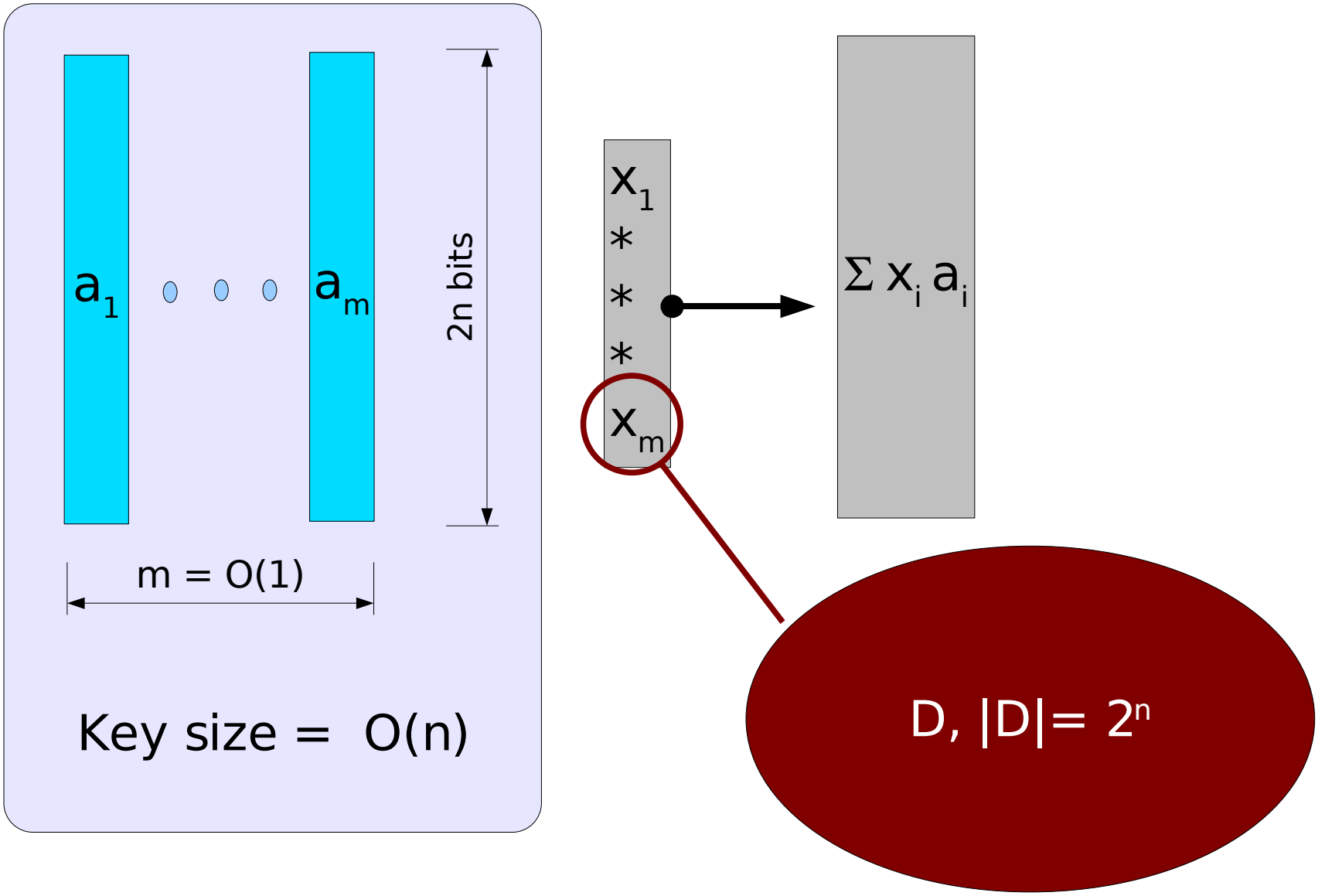
- Public key encryption [AD, Reg]
  - Requires planting a trapdoor for decryption
  - Can be done by using lattices where $\lambda_1 << \lambda_2$
- Unique SVP (uSVP)
  - Solve SVP on special class of lattices such that $\lambda_1 << \lambda_2$
  - Still worst-case assumption, but over smaller class of lattices
- [Rev] PKC from quantum hardness of SVP

Key Size of subset-sum function





Compact knapsacks



Ring choice and security

- Traditional compact knapsacks
  - a_i in Z_N, x_i in {0,...,M}, e.g. $M=2^n$ and $N=2^{2n}$
  - ILP with few vars: insecure!
- Quotient ring of polynomials:
  - a_i in $Z_p[X] / q(X)$, e.g.
  - x_i in $\{0,...,p^d\}^n$
- [Mic] If $q(X) = X^n - 1$, as hard to invert as solving SIVP on any cyclic lattice L, i.e.
  - L closed under [1,2,3,4,5] ----> [2,3,4,5,1]

More general: Ideal lattices

- q(X): monic polynomial in Z[X] of deg. n
  - R = Z[X] / q(X) is isomorphic to Z^n
  - h: $5X^2+3X-1$ ----> [5,3,-1]
- Ideal lattices
  - q(X) arbitrary monic polynomial
  - h(S) where S is an ideal of Z[X]/q(X)
  - If $q(X) = X^n - 1$, same as h(S) cyclic
- [PR,LM] Hash functions based on cyclic and ideal lattices with q(X) irreducible

Ideal lattices and small conjugates

- Two ways to map polynomials to vectors
  - Coefficients vector
  - Conjugates vector (Eval. at q-roots)
- Example
  - $q(X)=X^n+1$, $q(\omega_{2n}^{2k+1})=0$, $g(X)=X^3+2X^2+3X+4$
  - g(X) ----> [1,2,3,4]
  - g(X) ----> $[g(\omega_{2n}),g(\omega_{2n}^{3}),g(\omega_{2n}^{5}),g(\omega_{2n}^{7})]$
- Theorem:
  - $1/n^{1/2} < \max_k |g(\omega_{2n}^{2k+1})| / \max_k |g_k| < n^{1/2}$

Outline

- Introduction
  - Lattices and algorithms
  - Complexity and Cryptography
- Lattice based cryptography
  - Lattice based hash functions
  - Other cryptographic primitives
- Conclusion / Open Problems
  - Choosing security parameters
  - Using lattices with special properties

Setting security level

- Cryptographic functions as hard to break as solving $SIVP_{n \log(n)}$ in the worst-case
- What value of n should be used?
  - Large enough so no efficient algorithm solves SIVP on every n dimensional lattice
- How can we determine the worst-case complexity of SIVP?
  - Traditional challenge/cryptanalysis? Or not?

Method 1: worst-case challenge

- Traditional cryptanalysis
  - Designer picks random challenges
  - Cryptanalysts break challenges for money
  - Only appropriate for average-case assumptions
- “Worst-case” cryptanalysis
  - Designer picks worst possible input lattice
  - Problem: how can one find such worst lattices?

Method 2: algorithmic analysis

- Use worst-case analysis of best known algorithms
- Lot of recent interest and work
  - Worst-case examples for BKZ [Ajt]
  - Faster variants of LLL, etc. [Sch,NS,GHKN]
- Problem: too conservative?
  - Algorithms may perform better in practice than theoretical worst case
  - Heuristics: randomize basis vectors, etc.

Method 3: reverse challenge

- Cryptanalyst comes up with heuristic algorithm and claim on its performance
- Reverse challenge
  - The algorithm is the challenge!
  - Designer has to disprove the challenge by providing input lattice that results in bad performance
- Problem
  - ... socially unacceptable

Method 4: crypto challenge

- Forget about the proof of security
- Pick random instance of cryptographic function
  - E.g., random matrix A
- Cryptanalyst attack the challenge
  - E.g., find collision Ax=Ay
- Problem
  - Each application requires new cryptanalysis
  - Why proving security at all?

“Abstract” provable security

- Security proof as a qualitative statements
  - Attacks can be avoided by increasing security parameter
  - No conceptual security flaw in cryptographic function
  - Tell us what distribution should be used
- Use traditional cryptanalysis to determine suitable security parameters

Lattices with special structure

- Geometric structure (uSVP -> PKE)
  - E.g., $\lambda_1 << \lambda_2$
- Algebraic structure (CycSVP -> FFThash)
  - E.g., Rot(L) = L
- Are structured lattices easier than general ones?
  - Symplectic lattices are slightly easier [GHN]
  - Polytime approximation within poly(n)?
  - NP-hard in the worst-case?

Geometric Structure

- Evidence supporting hardness
  - Open problem: Given (random) lattice, decide if $\lambda_1 << \lambda_2$ or $\lambda_1 = \lambda_2$
- Evidence against
  - No NP-hardness result known
  - Cryptanalysis gives experimental evidence that uSVP is easier to some extent
- Open problems
  - Prove anything about uSVP

Algebraic Structure

- Evidence supporting hardness
  - Closely related to lattices arising in Algebraic Number Theory applications
  - ANT among first applications of LLL, still no substantially better specialized algorithm
  - LLL is “geometric”, it does not see algebraic structure
- LLL and Algebraic Number Theory
  - Applying LLL to ANT: great success story
  - It is time to apply ANT to lattice reduction!

Crypto from Algebraic Lattices Bibliography

- D.Micciancio: FOCS'02, Comp.Complexity (2007+, author's webpage)
- C.Peikert, A. Rosen: TCC'06
- V.Lyubashevsky, D.Micciancio: ICALP'06
- Lyubashevsky, Micciancio, Peikert, Rosen: NIST HASH '07 (mostly implementation)
- C.Peikert, A.Rosen: STOC '07
- NTRU (See Nick's paper)