From Ajtai-Dwork to NTRU:
The design of practical lattice based cryptosystems

Daniele Micciancio

University of California
San Diego

Acknowledgments

- [GGH97], [AD97], [HPS98]
- [M01] "Improving lattice based cryptosystems using the Hermite Normal Form", presented at CaLC 2001, (available at URL http://www.cse.ucsd.edu/~daniele)
- Conversations with Shai Halevi and Nick Howgrave-Graham

Hard Problems and Crypto

- Cryptography: design functions that are computationally hard to break (e.g., invert)
- Strategy:
  - Find a computationally hard problem P
  - Find a way to exploit this hardness to design functions that are as hard to invert as solving P
- Example:
  - Factoring problem: Given N=pq, find p and q
  - Rabin: x -> x*x mod N

The search of hard problems

- Hard problems are abundant in computer science (e.g., NP–complete problems)
- However, finding hard problems that are suitable for cryptographic applications is not easy:
  - Need problems that are hard on the average
  - Cryptographic applications require extra properties, e.g., a trapdoor to invert the function
  - E.g., Rabin: if p and q are known, then one can efficiently compute x given x*x mod (N=pq)

Candidate hard problems

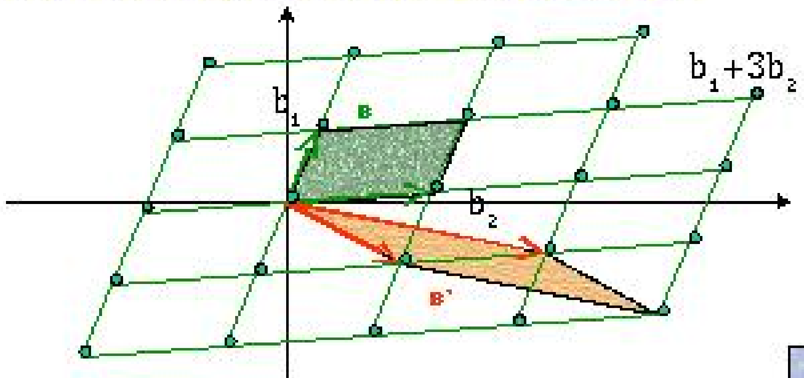
- Most hard problems currently used in cryptography are from number theory
- E.g., factoring, discrete logarithm
- Not desirable:
  - Evidence that most of these problems are not the hardest within NP
  - Breakthrough in number theory would be a disaster
  - Quantum computers can efficiently factor numbers

Lattice based cryptography

- Different class of problems to be used in crypto
- Many of these problems are NP–hard to solve exactly, or even approximately (within small factors) [vEB81, ABSS97, A96, M98, DKS98]
- Some lattice problems are provably hard on the average, assuming the worst case intractability of some other lattice problem [A97, CN97]
- No quantum algorithm is known

Lattices

- Set of all integer linear combinations of basis vectors **B**=$\{\mathbf{b}_1,\ldots,\mathbf{b}_n\}$
- Every lattice has infinitely many bases
- All bases have the same determinant



Lattice Cryptosystems

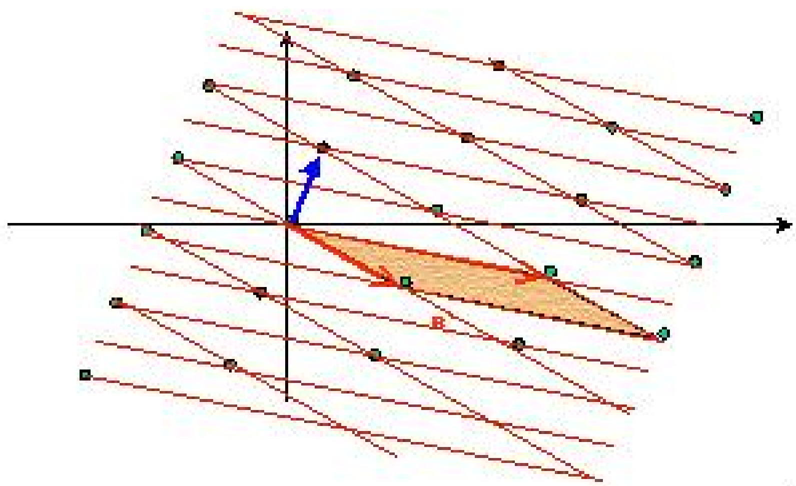
- Ajtai–Dwork cryptosystems [AD97]
- GGH cryptosystem [GGH97]
- NTRU [HPS98]
- Tensor Cryptosystems [FS99]
- HNF [M01]
- Other variants

This Talk

- Overview of main lattice based cryptosystems (AD1, GGH, NTRU)
- Technique to improve key size using HNF
- Appication to AD1, GGH, Tensor
- Comparison with NTRU
- Open problems

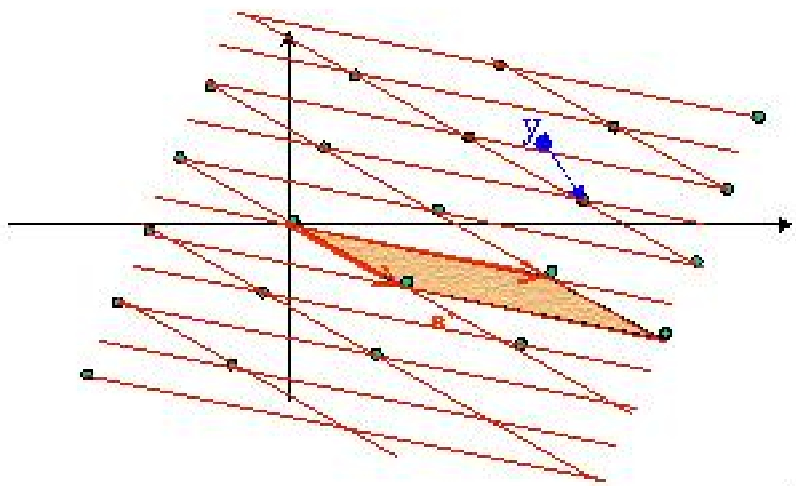
Shortest Vector Problem

- Given a lattice, find the nonzero lattice vector closest to the origin.



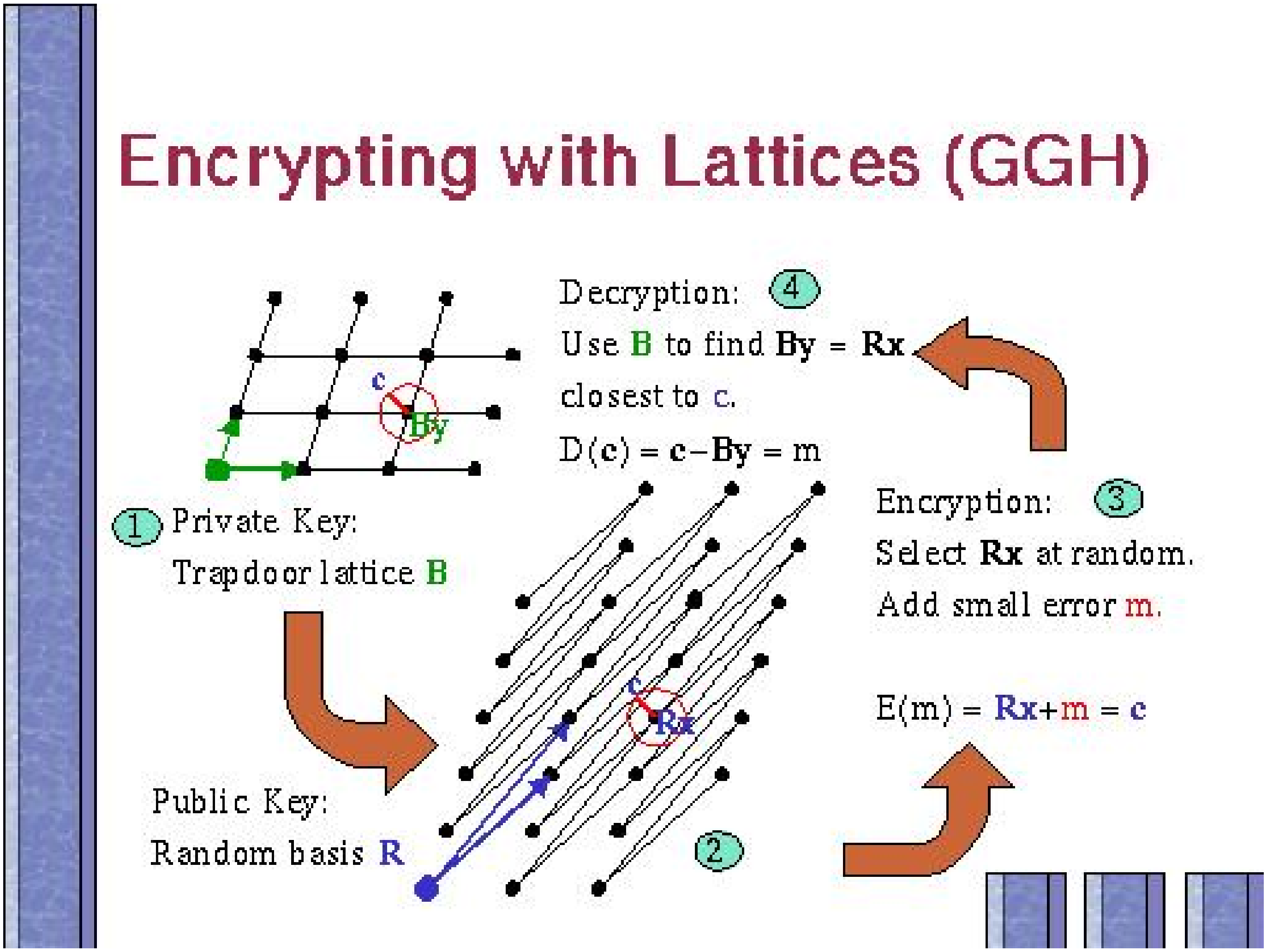
Closest vector problem

- Given a lattice B and a target point y, find the lattice point closest to the target



Other Hard Lattice Problems

- Problmes:
  - Shortest Basis (SBP): Given a lattice basis B, find smallest basis for the lattice L(B). [Several variants.]
  - Unique shortest vector problem (USVP): like SVP, but shortest vector is unique up to some polynomial factor
  - Covering radius problem: given a lattice L(B), find r such that every point in span(B) is within distance r from L(B)



Encrypting with Lattices (GGH)
Decryption: 4
Use B to find By = Rx
closest to c.
D(c) = c−By = m
c
By
1 Private Key:
Trapdoor lattice B
Encryption: 3
Select Rx at random.
Add small error m.
E(m) = Rx+m = c
c
Rx
Public Key:
Random basis R
2

Questions

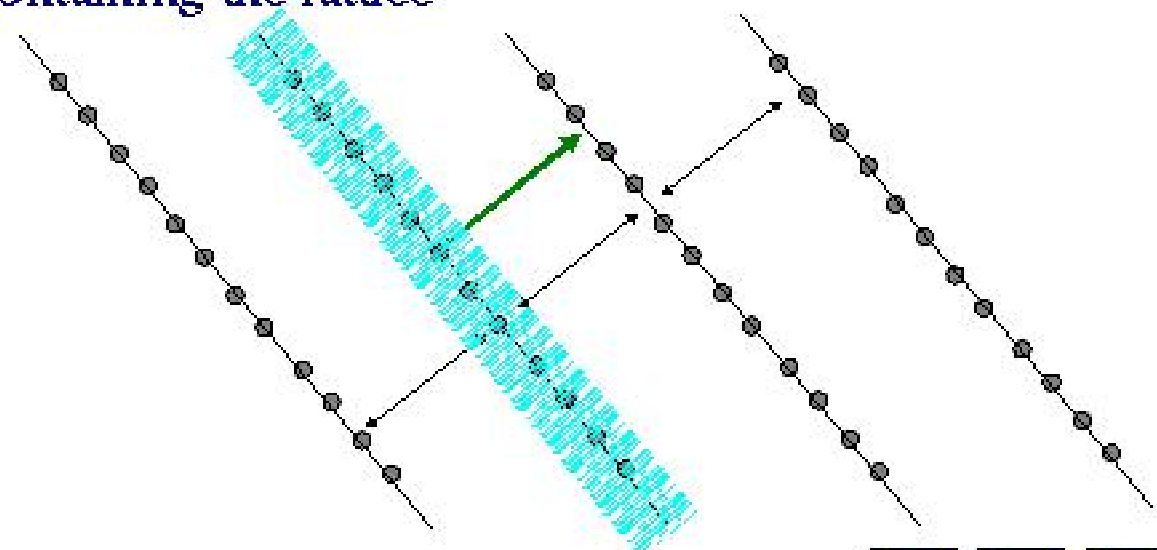
- How is the secret (good) basis chosen?
- How is the public basis computed from the private one?
- How is the public basis used to encrypt?
- How is the secret basis used to decrypt?

Secret basis and decryption

- Different cryptosystems suggest different ways to choose the secret basis
  - AD: short dual vector (or hidden hyperplane)
  - GGH: short lattice basis
  - Tensor: decomposition of the lattice
  - NTRU: short lattice vector
- The decryption algorithm depends on the choice of the secret basis

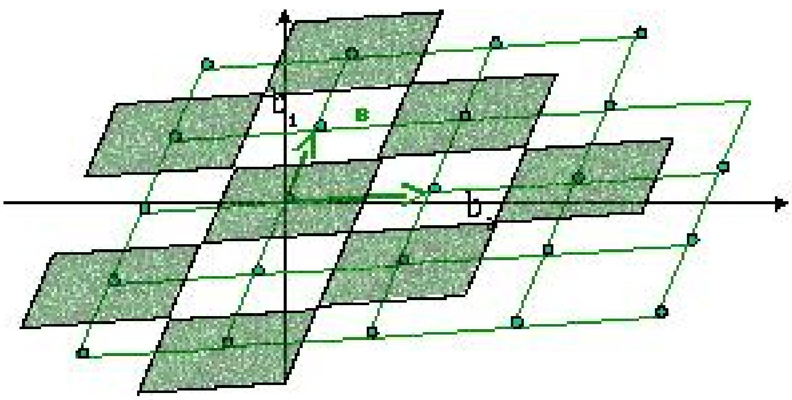
Ajtai–Dwork

- Trapdoor is a collection of hyperplanes containing the lattice



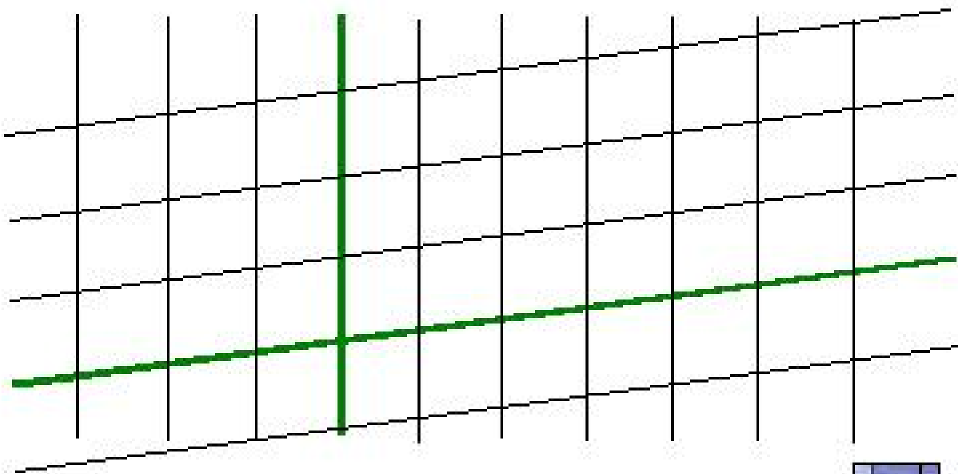
GGH

- Secrey key is a good (short, almost orthogonal) basis



Tensor based cryptosystem

- Trapdoor is a decomposition of the lattice as the tensor product of many small dimensional lattices



NTRU

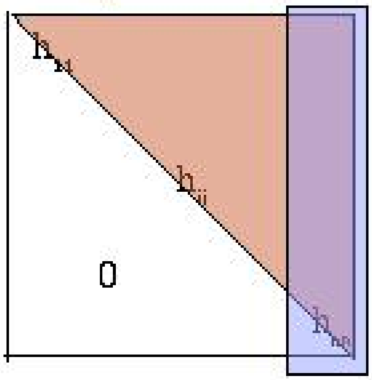
- Originally described as a cryptosystem based on polynomial ring arithmetics:
  - Secret key is a pair of polynomials f,g.
  - Public key is the quotiend $h=(g/f) \bmod (X^n-1,q)$
  - The encryption of message m using randomness r is the polynomial $c = 3hr + m \bmod q$
  - Decryption: $(fc \bmod (X^n-1,q)) / f \pmod{(X^n-1,3)}$

Choosing the public key

- Intuitive solution:
  - Apply a random transformation B >>> R
  - Method used in [GGH97, AD97, FS99]
- Analisys:
  - Integer lattices repeat identically when translated by multiples of det(B)
  - R has been properly randomized when all entries of R are roughly as big as the det(B)
  - Even if B is a 0–1 matrix, det(B) can be (n log n) bits, resulting in public keys of size (n^3 log n)

HNF Public Key

- Hermite Normal Form: Unique lower triangular matrix that generates the same lattice as B
- Every entry is reduced modulo the corresponding diagonal element

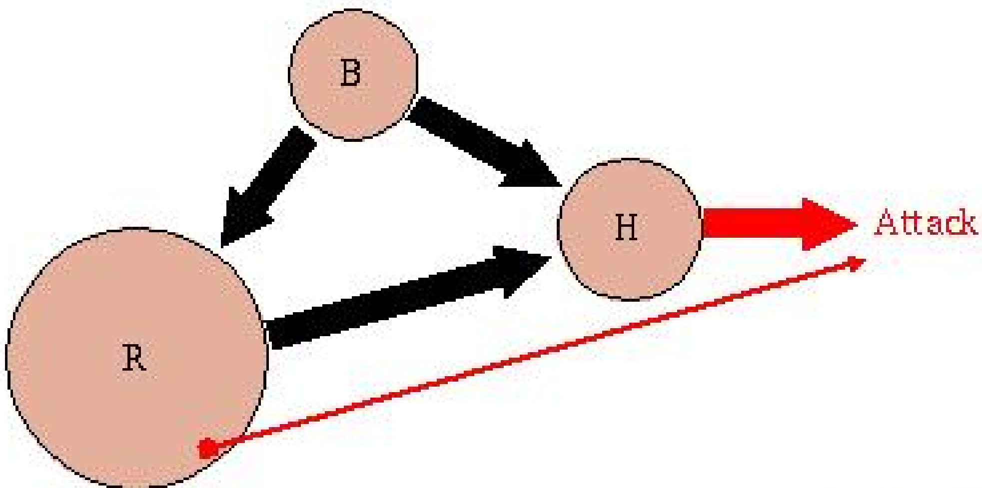


$h_{11} \ldots h_{nn} = \det(B)$

$\text{Size}(H) = n \ \text{size}(\det(B)) = n^2 \log n$

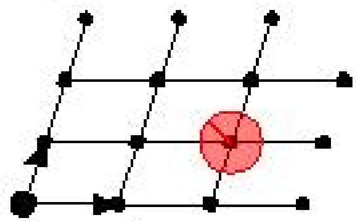
Security of HNF basis

- Since HNF can be efficiently computed from any other basis, it is the "most" secure basis

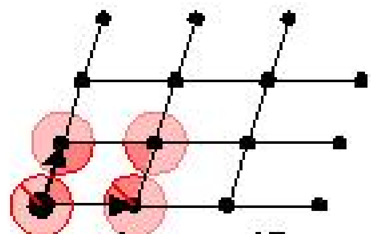


Encrypting with lattices

- Compute the chipertext as follows:
  - Instead of adding a random lattice vector Rx to m
  - Reduce m modulo the (orthogonalized) public basis
- Notice: c' = r(m) mod R = c mod R can be computed from c, therefore it is more secure.

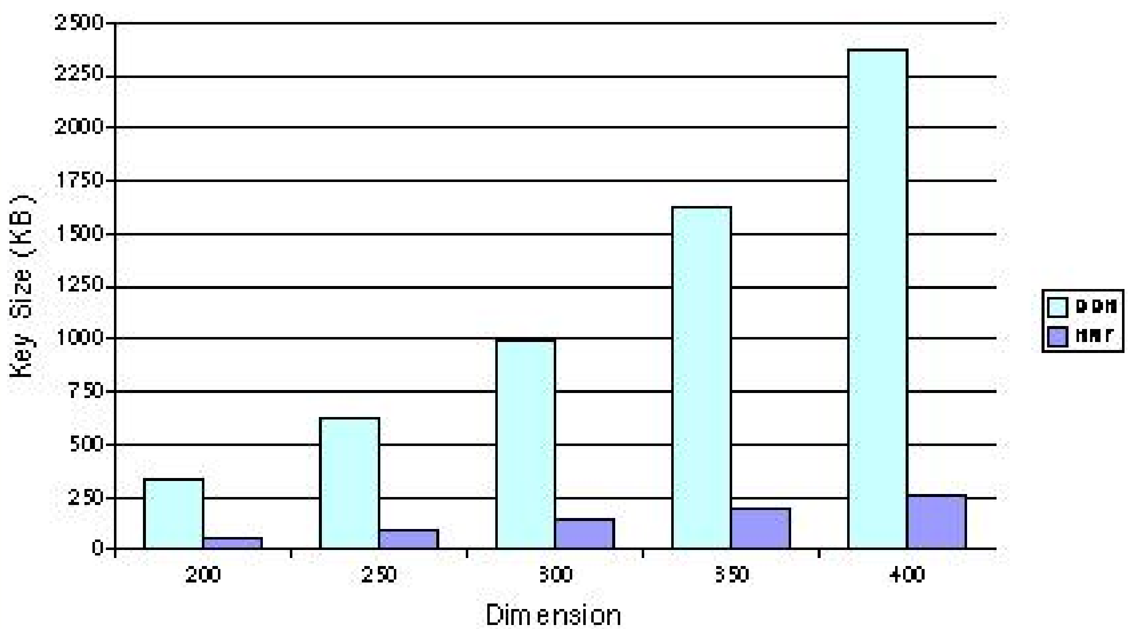


c = Rx+m

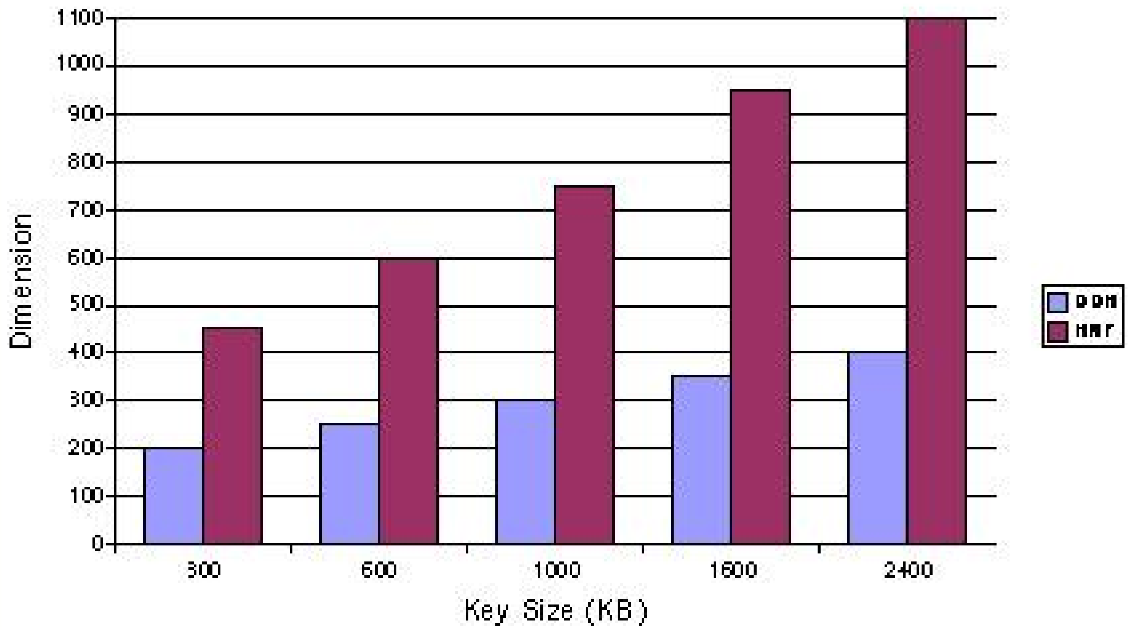


c' = m mod R

Security vs Key Size



Key size vs Security



Lesson

- The "right" choice is better than random choice
  - HNF basis is at least as secure as any other basis
  - HNF basis is much smaller than random basis
- The modified "cryptosystems" are deterministic
  - They should be regarded as trapdoor functions
  - Can be transformed into encryption schemes
- Despite the reduction, key size is still much bigger than RSA, Rabin, etc.

Optimality of HNF key size

- Simple counting argument shows that the bit size of HNF basis is optimal: there are exp(s) different lattices with HNF of size s!
- In order to get smaller key size, one need to use lattices of special form
- What kind of lattices can be used to reduce the public key size?

Modular lattices

- L is q–modular if x=0 mod q implies x is in L
- The public key size get slighly smaller: instead of $O(n^2 \log n)$, now is $O(n^2 \log q)$
- Still nor enough. Even for q=2, there are still $\exp(O(n^2))$ different lattices:
  - Consider all triangular matrices with 2 on the diagonal, and 0/1 off the diagonal
  - There are $2^{(n(n-1)/2)}$ such matrices, and they all represent different lattices.

Cyclic lattices

- For any $x=[x_1,\ldots,x_n]$, define the cyclic shift $rot(x)=[x_2,x_3,\ldots,x_n,x_1]$
- A lattice is cyclic if (x in L) implies (rot(x) in L)
- Given x, the smallest cyclic lattice containing x is generated by x, rot(x),..., $rot^n(x)$.
- For most vectors, x, rot(x),..., $rot^n(x)$ are linearly independent, and C(x) is full dimentional
- There are many cyclic lattices that can be represented by a single vector x.

2–cyclic lattices

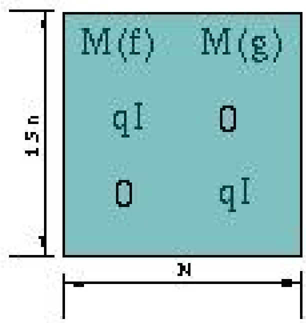
- Assume n is even, and define the double rotation $rot_2([x,y]) = [rot(x),rot(y)]$.
- A lattice is **2**–cyclic if ([x,y] in L) implies ($rot_2([x,y])$ in L)
- Notice that $rot_2^{n/2}([x,y]) = [x,y]$
- Therefore, the **2**–cyclic lattice generated by a single vector is never full dimensional!

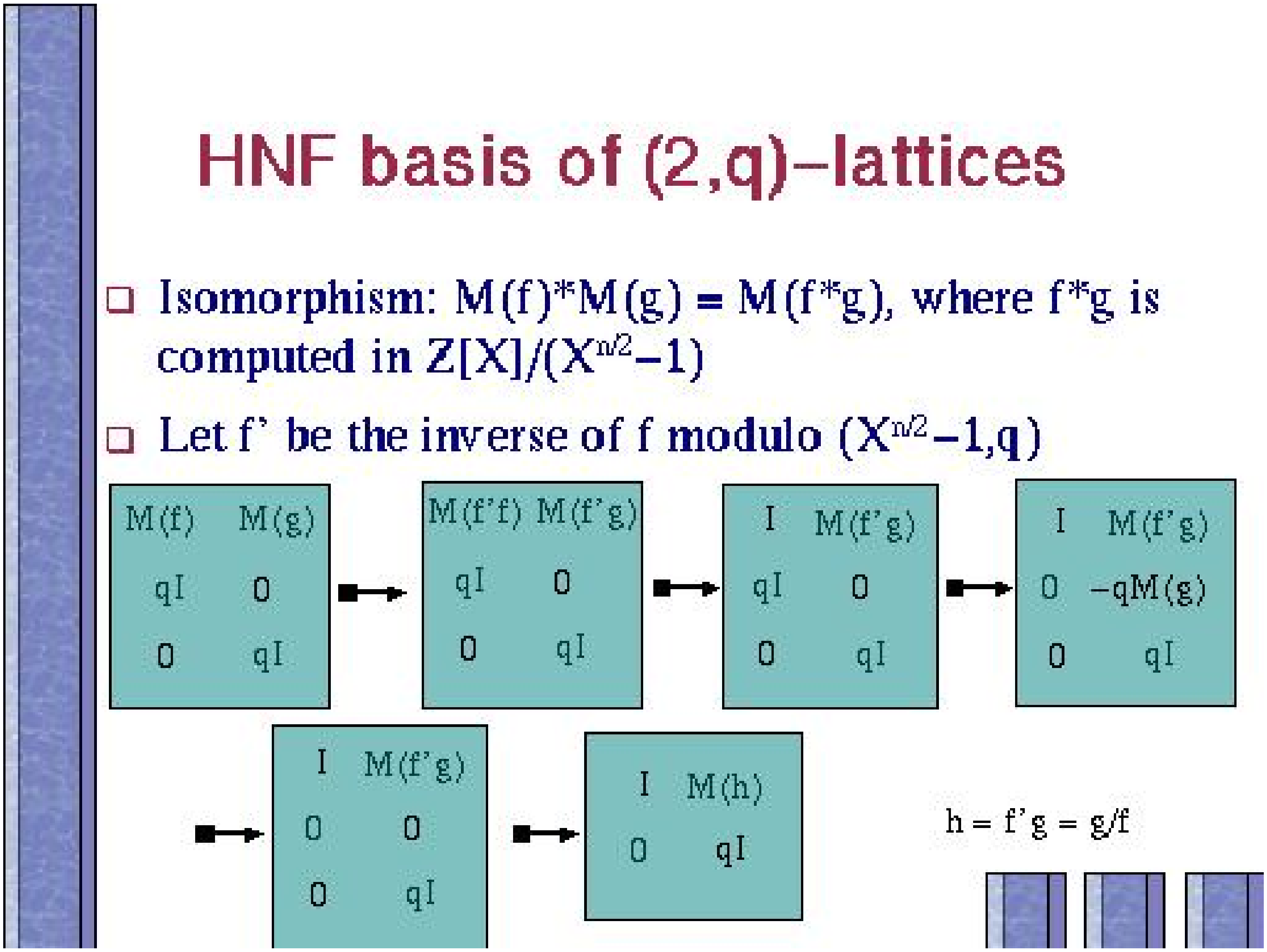
2–cyclic q–modular lattices

- Consider any vector [x,y]
- The smallest 2–cyclic q–modular lattice containing [x,y] is generated by vectors
  - [x,y], rot_2([x,y]), ..., $rot_2^{(n/2)-1}$([x,y])
  - All n vectors (0,...,q,...,0)
- These are (3/2)n vectors in n dimentional space, so they are certainly linearly dependent
  - A basis can be computed using HNF algorithm
  - The lattice is always full rank

Generating (2,q)–lattices

- Let [f,g] be a short vector, and let L be the (2,q)–lattice generated by [f,g].
- Let M(f) be the circulant matrix associate to f, i.e., the square matrix with rows $rot^i(f)$
- L is generated by the rows of




HNF basis of (2,q)-lattices
Isomorphism: M(f)*M(g) = M(f*g), where f*g is computed in Z[X]/(X^{n/2}-1)
Let f' be the inverse of f modulo (X^{n/2}-1,q)
M(f) M(g)
qI 0
0 qI
M(f'f) M(f'g)
qI 0
0 qI
I M(f'g)
qI 0
0 qI
I M(f'g)
0 -qM(g)
0 qI
I M(f'g)
0 0
0 qI
I M(h)
0 qI
h = f'g = g/f

Encrypting with (2,q)-lattices

- We are given the HNF basis H, and a small error vector [s,m], and want to compute [s,m] mod H
- Morphism: $s*M(f) = s*f \mod (X^{n/2} - 1)$

$$H = \begin{bmatrix} M(1) & M(h) \\ 0 & qI \end{bmatrix}$$

$$\begin{matrix} [\ s & m\ ] \\ [\ 0 & m-s*M(h)\ ] \\ [\ 0 & m-s*h\ ] \\ [\ 0 & e\ ] \end{matrix}$$

$$e = (m - s*h) \mod (X^{n/2} - 1, q)$$

$$s = -3r \qquad e = (3r*h + m) \mod (X^{n/2} - 1, q)$$

NTRU, Alternative definition

- Secret key: short lattice vector [f,g]
- Public key: HNF basis of the smallest 2–cyclic q-modular lattice containing [f,g].
- "Encryption": input is a short error vector of the form x = [–3s, m]. Output is (x mod H).
- Decryption: ???

Conclusion

- HNF technique gives an optimal way to compute public basis for lattices. In particular, HNF can be used to improve [AD1, GGH, FS].
- HNF public basis requires $O(n^2)$ bits in general. In order to get shorter keys, one has to consider special classes of lattices
- NTRU is an interesting example of HNF cryptosystem, when applied to (2,q)–lattices.

Open problems (1)

- Find other classes of lattices that result in O(n) public key size. E.g., can we do encryption using cyclic lattices?
- Complexity of cyclic or (2,q) lattices:
  - Are SVP, CVP NP–hard?
  - Is CVP with preprocessing hard?
- Is there a natural geometric interpretation for NTRU decryption procedure?
- Is there some general technique that can be used for decryption?

Open Problems (2)

- Ajtai–Dwork proposed also a cryptosystem AD2 with worst–case/average–case connection. Can the HNF technique be adapted to work on AD2?
- Average–case/worst–case connection for cyclic lattices a la' Ajtai. (YES! [M02] gives efficient OWF based on worst case hardness of apprixmating SVP in cyclic lattices)