Pseudorandom Knapsacks and the Sample Complexity of LWE Search-to-Decision Reductions

Crypto 2011

Daniele Micciancio
Petros Mol

UCSDCSE
Computer Science and Engineering

August 17, 2011

Learning With Errors (**LWE**)

secret
$\mathbf{s} = (s_1, \ldots, s_n) \in \mathbb{Z}_q^n$

public: integers n, q

e_i small error from a known distribution

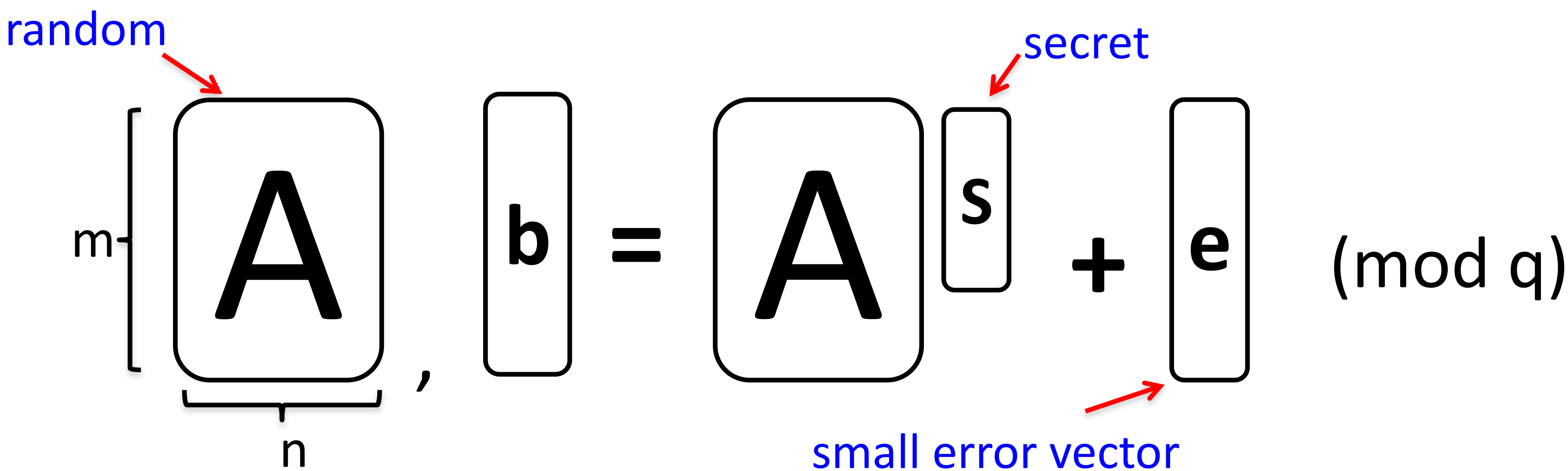
$(\mathbf{a}_1, b_1)$ $(\mathbf{a}_2, b_2)$ … $(\mathbf{a}_m, b_m)$

$\mathbf{a}_i \in_R \mathbb{Z}_q^n$

noise

$b_i = \mathbf{a}_i \cdot \mathbf{s} + e_i \mod q$

Goal: Find **s**



LWE Background

- Introduced by Regev [R05]
- q = 2, Bernoulli noise -> Learning Parity with Noise (LPN)
- Extremely successful in Cryptography
  - IND-CPA Public Key Encryption [Regev05]
  - Injective Trapdoor Functions/ IND-CCA encryption [PW08]
  - Strongly Unforgeable Signatures [GPV08, CHKP10]
  - (Hierarchical) Identity Based Encryption [GPV08, CHKP10, ABB10]
  - Circular- Secure Encryption [ACPS09]
  - Leakage-Resilient Cryptography [AGV09, DGK+10, GKPV10]
  - (Fully) Homomorphic Encryption [GHV10, BV11b]

LWE: Search & Decision

Public parameters

n: size of the secret, m: #samples
q: modulus, χ :error distribution

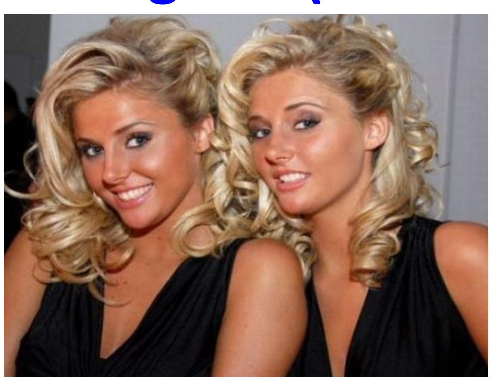
Find (Search)

$\mathbf{A} \in_R \mathbb{Z}_q^{m \times n}$

Given: $(\mathbf{A}, \mathbf{A} \cdot \mathbf{s} + \mathbf{e}) \quad \mathbf{e} \sim \chi^m$

Goal: find **s** (or **e**)

Distinguish (Decision)



Given: $(\mathbf{A}, \mathbf{t} \in \mathbb{Z}_q^m)$

Goal: decide if $\mathbf{t} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$

or $\mathbf{t} \in_R \mathbb{Z}_q^m$

Search-to-Decision reductions (S-to-D)

Why do we care?

decision problems

- all LWE-based constructions rely on decisional LWE
- strong indistinguishability flavor of security definitions

search problems

- their hardness is better understood

Search-to-Decision reductions (S-to-D)

Why do we care?

decision problems

search problems

- all LWE-based constructions rely on decisional LWE
- strong indistinguishability flavor of security definitions

- their hardness is better understood

- S-to-D reductions: "Primitive Π is ABC-Secure assuming **search problem** P is hard"

Our results

- Toolset for studying Search-to-Decision reductions for LWE with **polynomially bounded noise**.
  - Subsume and extend previously known ones
  - Reductions are in addition **sample-preserving**

- Powerful and usable criteria to establish Search-to-Decision equivalence for general classes of **knapsack functions**

- Use known techniques from Fourier analysis in a new context. Ideas potentially useful elsewhere

Our results

- Toolset for studying Search-to-Decision reductions for LWE with **polynomially bounded noise**.
  - Subsume and extend previously known ones
  - Reductions are in addition **sample-preserving**

- Powerful and usable criteria to establish Search-to-Decision equivalence for general classes of **knapsack functions**

- Use known techniques from Fourier analysis in a new context. Ideas potentially useful elsewhere

Bounded knapsack functions over groups

Parameters

- integer m
- **finite** abelian group G
- set S = {0,…, s - 1} of integers, s: **poly(m)**

(Random) Knapsack family $S^m \longrightarrow G$

Sampling $\mathbf{g} = (g_1, \ldots, g_m)$ where $g_i \in_R G$

Evaluation $\mathbf{g}(\mathbf{x}) = \mathbf{g} \cdot \mathbf{x} = \sum_{i=1}^{m} x_i g_i \in G$

Example

(random) modular subset sum: $S = \{0, 1\},\ G = \mathbb{Z}_M$

Knapsack functions: Computational problems

$\mathcal{D}$ distribution over S^m $\quad (G, \mathcal{D})$ **public**

invert (search)	**Input:** $\mathbf{g},\ y = \mathbf{g} \cdot \mathbf{x} \quad (g_i \in_R G,\ \mathbf{x} \sim \mathcal{D})$ **Goal:** Find **x**

Distinguish (decision)	**Input:** Samples from either: $\mathcal{F}_{\mathcal{D}} = (\mathbf{g}, \mathbf{g} \cdot \mathbf{x}) \ \ (g_i \in_R G, \mathbf{x} \sim \mathcal{D})$ $\mathcal{F}_{\mathcal{U}} = (\mathbf{g}, u) \ \ (g_i \in_R G, u \in_R G)$ **Goal:** Label the samples

Notation: $\mathcal{K}(G, \mathcal{D})$ family of knapsacks over G with distribution $\mathcal{D}$

Glossary: If decision problem is hard, function is **pseudorandom** (PRG)
If search problem is hard, function is **One-Way**

Search-to-Decision: Known results

Decision as hard as search when…

[Impagliazzo, Naor 89] : (random) modular subset sum

$G = \mathbb{Z}_M$, cyclic group

$\mathcal{D}$ **uniform** over $S^m = \{0,1\}^m$

[Fischer, Stern 96]: syndrome decoding

$G = \mathbb{Z}_2^k$, vector group

$\mathcal{D}$ **uniform** over all m-bit vectors with Hamming weight w.

Our contribution: S-to-D for general knapsack

$\mathcal{K}(G, \mathcal{D})$: knapsack family with range G and input distribution $\mathcal{D}$ over $\{0, \ldots, s-1\}^m$ s: **poly(m)**

$\mathcal{K}(G, \mathcal{D})$ One-Way

$\boxed{\mathcal{K}(G/dG, \mathcal{D}) \text{ PRG } \forall d < s}$

$\mathcal{K}(G, \mathcal{D})$ PRG

Our contribution: S-to-D for general knapsack

$\mathcal{K}(G, \mathcal{D})$: knapsack family with range G and input distribution $\mathcal{D}$ over $\{0, \ldots, s-1\}^m$ s: **poly(m)**

Main Theorem

$\mathcal{K}(G, \mathcal{D})$ One-Way + $\mathcal{K}(G/dG, \mathcal{D})$ PRG $\forall d < s$ $\Rightarrow$ $\mathcal{K}(G, \mathcal{D})$ PRG

Our contribution: S-to-D for general knapsack

Main Theorem

$\mathcal{K}(G,\mathcal{D})$ One-Way $+$ $\mathcal{K}(G/dG,\mathcal{D})$ PRG $\forall d < s$ $\Rightarrow$ $\mathcal{K}(G,\mathcal{D})$ PRG

$\mathcal{K}(G/dG,\mathcal{D})$ PRG $\forall d < s$

Much less restrictive than it seems

In most interesting cases holds in a strong **information theoretic** sense

S-to-D for general knapsack: Examples

$\mathcal{K}(G,\mathcal{D})$ One-Way $\Rightarrow$ $\mathcal{K}(G,\mathcal{D})$ PRG

Subsumes
[IN89,FS96]
and more

Any group G and **any distribution** over $\{0,1\}^m$

Any group G with **prime exponent** and **any distribution**

And many more…

using known information theoretical tools (LHL, entropy bounds etc)

Proof Sketch

Inverter

Input: **g** , **g**·**x**
Goal: Find **x**

Distinguisher

Input: $\mathcal{F}_{\mathcal{D}}$ or $\mathcal{F}_{\mathcal{U}}$
Goal: Distinguish

Reminder

$$\mathcal{F}_{\mathcal{D}} = (\mathbf{g}, \mathbf{g} \cdot \mathbf{x}) \;\; (g_i \in_R, \mathbf{x} \sim \mathcal{D})$$
$$\mathcal{F}_{\mathcal{U}} = (\mathbf{g}, u) \;\; (g_i \in_R, u \in_R G)$$

Proof Sketch

Inverter <= step 1 Predictor <= step 2 Distinguisher

Input: **g** , **g·x**
Goal: Find **x**

Input: **g** , **g·x**, **r**
Goal: find **x·r** (mod t)

Input: $\mathcal{F}_{\mathcal{D}}$ or $\mathcal{F}_{\mathcal{U}}$
Goal: Distinguish

Proof follows outline of [IN89]

Step 1: Goldreich–Levin replaced by **general conditions** for inverting given noisy predictions for **x·r** (mod t) for possibly composite t

-Tool: learning heavy Fourier coefficients of general functions [AGS03]

Step 2: Given a distinguisher, we get a predictor satisfying general conditions of step 1.

Proof significantly more involved than [IN89]

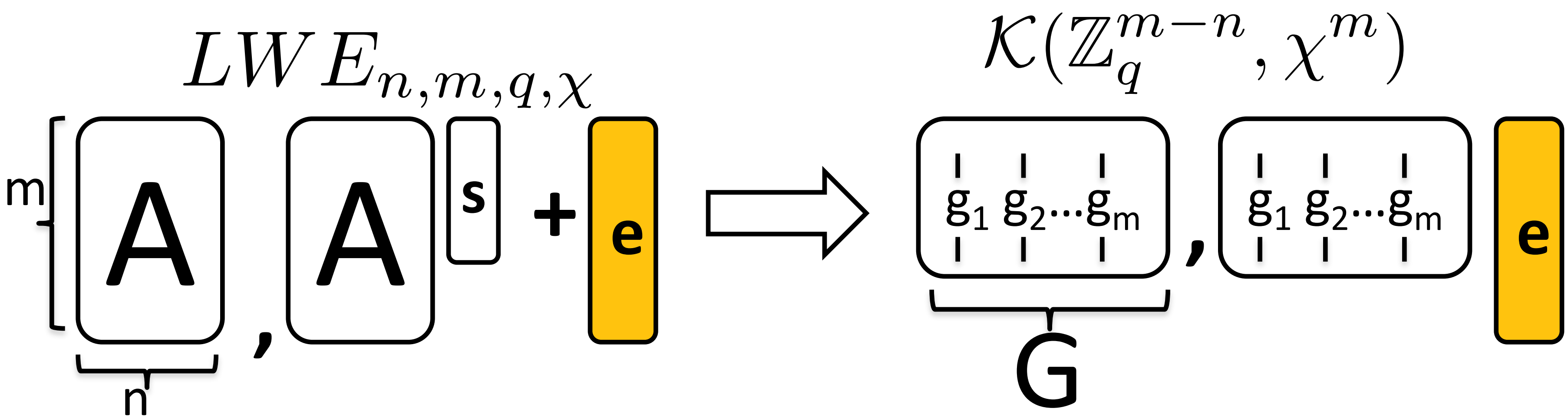
Our results

- Toolset for studying Search-to-Decision reductions for LWE with **polynomially bounded noise**.
  - Subsume and extend previously known ones
  - Reductions are in addition **sample-preserving**

- Powerful and usable criteria to establish Search-to-Decision equivalence for general classes of **knapsack functions**

- Use known techniques from Fourier analysis in a new context. Ideas potentially useful elsewhere

What about LWE?



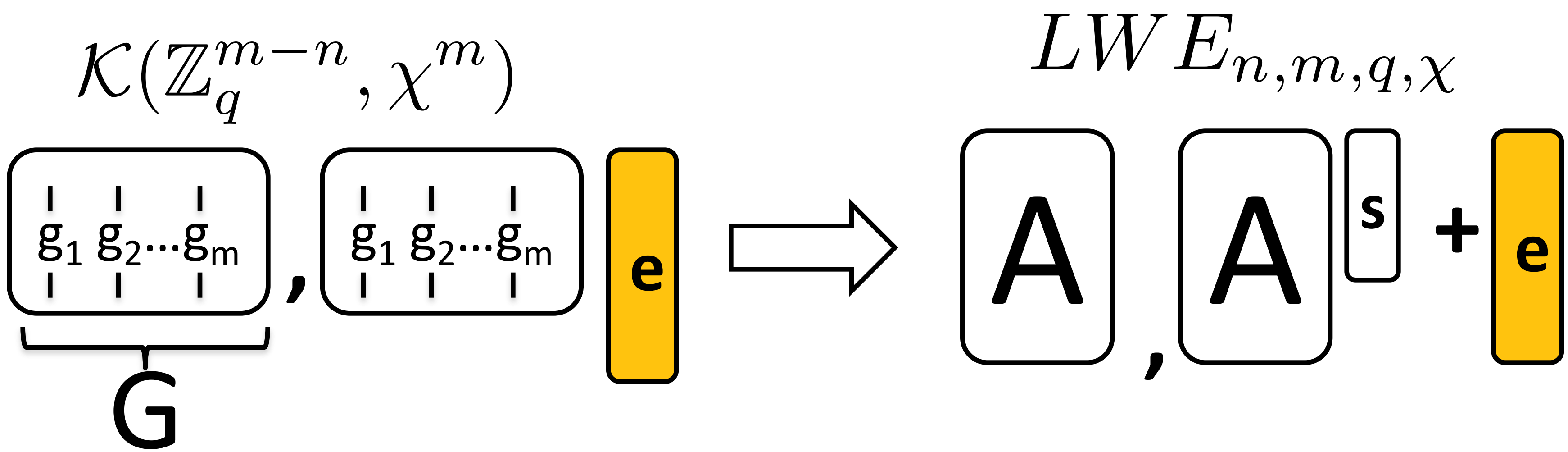
G is the parity check matrix for the code generated by **A**

$$\mathbf{G} \cdot \mathbf{A} = \mathbf{0} \pmod q$$

Error **e** from LWE → unknown input of the knapsack

If **A** is "random", **G** is also "random"

What about LWE?



The transformation works in the other direction as well

Putting all the pieces together…

Search	Search		Decision	Decision

(A, As +e) <= (G, Ge) <= **(G’, G’e) <= (A’,A’s’ + e)**

S-to-D for knapsack

LWE Implications

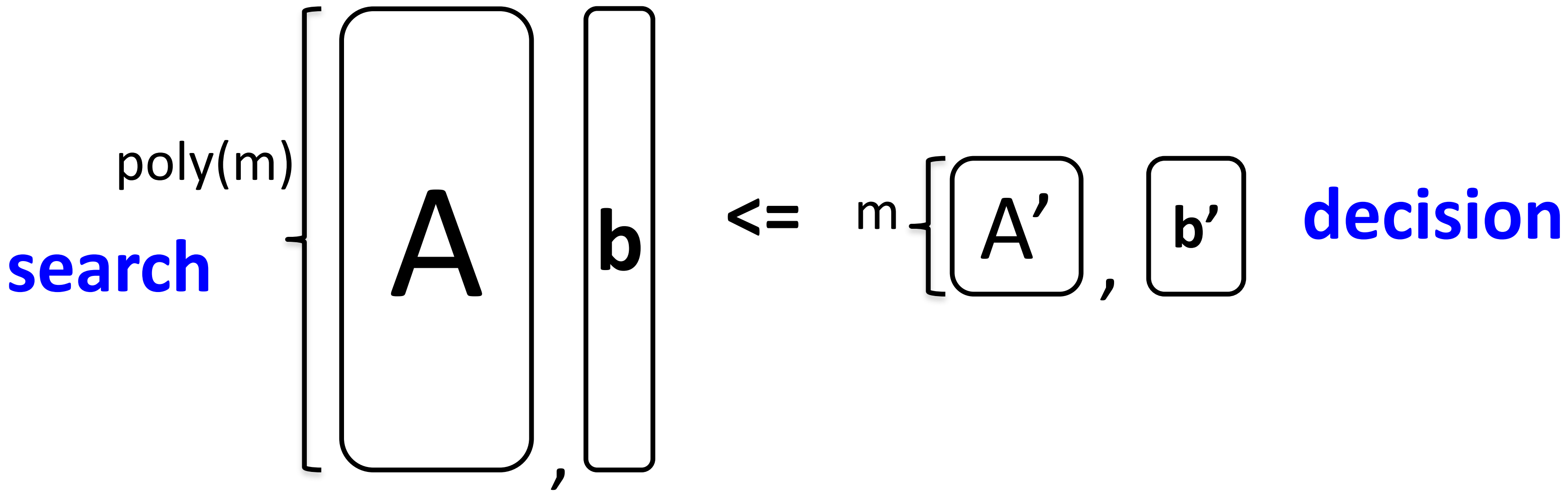
LWE reductions follow from knapsacks reductions over $\mathbb{Z}_q^{m-n}$

All known Search-to-Decision results for **LWE/LPN** with bounded error [BFKL93, R05, ACPS09, KSS10] follow as a direct corollary

Search-to-Decision for new instantiations of LWE

LWE: Sample Preserving S-to-D

Previous reductions



Ours: **sample-preserving**

If we can solve decision LWE given **m** samples, we can solve search LWE given **m** samples

Caveat: Inverting probability goes down (seems unavoidable)

Why care about #samples?

- LWE-based schemes often expose a certain number of samples, say **m**

- With sample-preserving S-to-D we can base their security on the hardness of search LWE with **m** samples

- Concrete algorithmic attacks against LWE [MR09, AG11] are sensitive to the number of exposed samples
  - for some parameters, LWE is completely broken by [AG11] if number of given samples above a certain threshold

Open problems

Sample preserving reductions for

1. LWE with unbounded noise

   - used in various settings [Pei09, GKPV10, BV11b, BPR11]
   - some reductions known [Pei09] but not sample-preserving

2. ring LWE

   - Samples (a, a*s+e) where a, s, e drawn from $\mathbf{R}=Z_q[x]/<f(x)>$
   - non sample-preserving reductions known [LPR10]