Linear Algebra with Errors: On the Complexity of the Learning with Errors Problem

Martin R. Albrecht

joint work with C. Cid, J-C. Faugère, R. Fitzpatrick, and L. Perret

SIAM AG'13

Contents

Learning with Errors

Definition (Learning with Errors)

Let $n \geq 1$, $m \gg n$, q odd, χ be a probability distribution on $\mathbb{Z}_q$ and $\mathbf{s}$ be a secret vector in $\mathbb{Z}_q^n$.

Let $\mathbf{e} \leftarrow_\$ \chi^m$, $\mathbf{A} \leftarrow_\$ \mathcal{U}(\mathbb{Z}_q^{m \times n})$. We denote by $L_{\mathbf{s},\chi}^{(n)}$ the distribution on $\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$ produced as $(\mathbf{A}, \mathbf{A} \cdot \mathbf{s} + \mathbf{e})$.

Decision-LWE is the problem of deciding whether $\mathbf{A}, \mathbf{c} \leftarrow_\$ \mathcal{U}(\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m)$ or $\mathbf{A}, \mathbf{c} \leftarrow_\$ L_{\mathbf{s},\chi}^{(n)}$.

In other words: Is $\mathbf{c}$ sampled uniformly randomly or is it $\mathbf{A} \cdot \mathbf{s} + \mathbf{e}$ where typically $\mathbf{e}$ is "small".

Typically, χ is a discrete Gaussian distribution with small standard deviation.

Learning with Errors

Definition (Learning with Errors)

Let $n \geq 1$, $m \gg n$, q odd, χ be a probability distribution on $\mathbb{Z}_q$ and $\mathbf{s}$ be a secret vector in $\mathbb{Z}_q^n$.

Let $\mathbf{e} \leftarrow_\$ \chi^m$, $\mathbf{A} \leftarrow_\$ \mathcal{U}(\mathbb{Z}_q^{m \times n})$. We denote by $L_{\mathbf{s},\chi}^{(n)}$ the distribution on $\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$ produced as $(\mathbf{A}, \mathbf{A} \cdot \mathbf{s} + \mathbf{e})$.

Decision-LWE is the problem of deciding whether $\mathbf{A}, \mathbf{c} \leftarrow_\$ \mathcal{U}(\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m)$ or $\mathbf{A}, \mathbf{c} \leftarrow_\$ L_{\mathbf{s},\chi}^{(n)}$.

In other words: Is $\mathbf{c}$ sampled uniformly randomly or is it $\mathbf{A} \cdot \mathbf{s} + \mathbf{e}$ where typically $\mathbf{e}$ is "small".

Typically, χ is a discrete Gaussian distribution with small standard deviation.

Learning with Errors

Definition (Learning with Errors)

Let $n \geq 1$, $m \gg n$, q odd, χ be a probability distribution on $\mathbb{Z}_q$ and $\mathbf{s}$ be a secret vector in $\mathbb{Z}_q^n$.

Let $\mathbf{e} \leftarrow_\$ \chi^m$, $\mathbf{A} \leftarrow_\$ \mathcal{U}(\mathbb{Z}_q^{m \times n})$. We denote by $L_{\mathbf{s},\chi}^{(n)}$ the distribution on $\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$ produced as $(\mathbf{A}, \mathbf{A} \cdot \mathbf{s} + \mathbf{e})$.

Decision-LWE is the problem of deciding whether $\mathbf{A}, \mathbf{c} \leftarrow_\$ \mathcal{U}(\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m)$ or $\mathbf{A}, \mathbf{c} \leftarrow_\$ L_{\mathbf{s},\chi}^{(n)}$.

In other words: Is $\mathbf{c}$ sampled uniformly randomly or is it $\mathbf{A} \cdot \mathbf{s} + \mathbf{e}$ where typically $\mathbf{e}$ is "small".

Typically, χ is a discrete Gaussian distribution with small standard deviation.

Learning with Errors

Definition (Learning with Errors)

Let $n \geq 1$, $m \gg n$, q odd, χ be a probability distribution on $\mathbb{Z}_q$ and $\mathbf{s}$ be a secret vector in $\mathbb{Z}_q^n$.

Let $\mathbf{e} \leftarrow_\$ \chi^m$, $\mathbf{A} \leftarrow_\$ \mathcal{U}(\mathbb{Z}_q^{m \times n})$. We denote by $L_{\mathbf{s},\chi}^{(n)}$ the distribution on $\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m$ produced as $(\mathbf{A}, \mathbf{A} \cdot \mathbf{s} + \mathbf{e})$.

Decision-LWE is the problem of deciding whether $\mathbf{A}, \mathbf{c} \leftarrow_\$ \mathcal{U}(\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m)$ or $\mathbf{A}, \mathbf{c} \leftarrow_\$ L_{\mathbf{s},\chi}^{(n)}$.

In other words: Is $\mathbf{c}$ sampled uniformly randomly or is it $\mathbf{A} \cdot \mathbf{s} + \mathbf{e}$ where typically $\mathbf{e}$ is "small".

Typically, χ is a discrete Gaussian distribution with small standard deviation.

Applications

- Public-Key Encryption, Digital Signature Schemes
- Identity-based Encryption: encrypting to an identity (e-mail address ...) instead of key
- Fully-homomorphic encryption: computing with encrypted data
- ...

Asymptotic Security

Reduction of worst-case hard lattice problems such as Closest Vector Problem (CVP) to average-case LWE.

But to build cryptosystems we need to understand the hardness of concrete instances: Given m, n, q and χ how many operations does it take to solve Decision-LWE?

Asymptotic Security

Reduction of worst-case hard lattice problems such as Closest Vector Problem (CVP) to average-case LWE.

But to build cryptosystems we need to understand the hardness of concrete instances: Given m, n, q and χ how many operations does it take to solve Decision-LWE?

Solving Strategies

Given $\mathbf{A}, \mathbf{c}$ with $\mathbf{c} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$ solve the problem in the primal lattice or the dual lattice.

- Solve the Bounded-Distance Decoding (BDD) problem in the primal lattice: Find $\mathbf{s}'$ such that

$$\|\mathbf{y} - \mathbf{c}\| \text{ is minimised, for } \mathbf{y} = \mathbf{A} \cdot \mathbf{s}'.$$

- Solve the Short-Integer-Solutions (SIS) problem in the scaled dual lattice. Find a short $\mathbf{y}$ such that

$$\mathbf{y} \cdot \mathbf{A} = 0 \text{ and check if } \langle \mathbf{y}, \mathbf{c} \rangle = \mathbf{y} \cdot (\mathbf{A} \cdot \mathbf{s} + \mathbf{e}) = \langle \mathbf{y}, \mathbf{e} \rangle \text{ is short.}$$

In this talk

solving SIS using combinatorial techniques and

no bound on m.

Solving Strategies

Given $\mathbf{A}, \mathbf{c}$ with $\mathbf{c} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$ solve the problem in the primal lattice or the dual lattice.

- Solve the Bounded-Distance Decoding (BDD) problem in the primal lattice: Find $\mathbf{s}'$ such that

$$\|\mathbf{y} - \mathbf{c}\| \text{ is minimised, for } \mathbf{y} = \mathbf{A} \cdot \mathbf{s}'.$$

- Solve the Short-Integer-Solutions (SIS) problem in the scaled dual lattice. Find a short $\mathbf{y}$ such that

$$\mathbf{y} \cdot \mathbf{A} = 0 \text{ and check if } \langle \mathbf{y}, \mathbf{c} \rangle = \mathbf{y} \cdot (\mathbf{A} \cdot \mathbf{s} + \mathbf{e}) = \langle \mathbf{y}, \mathbf{e} \rangle \text{ is short.}$$

In this talk

solving SIS using combinatorial techniques and

no bound on m.

Solving Strategies

Given $\mathbf{A}, \mathbf{c}$ with $\mathbf{c} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$ solve the problem in the primal lattice or the dual lattice.

- Solve the Bounded-Distance Decoding (BDD) problem in the primal lattice: Find $\mathbf{s}'$ such that

$$\|\mathbf{y} - \mathbf{c}\| \text{ is minimised, for } \mathbf{y} = \mathbf{A} \cdot \mathbf{s}'.$$

- Solve the Short-Integer-Solutions (SIS) problem in the scaled dual lattice. Find a short $\mathbf{y}$ such that

$$\mathbf{y} \cdot \mathbf{A} = 0 \text{ and check if } \langle \mathbf{y}, \mathbf{c} \rangle = \mathbf{y} \cdot (\mathbf{A} \cdot \mathbf{s} + \mathbf{e}) = \langle \mathbf{y}, \mathbf{e} \rangle \text{ is short.}$$

In this talk

solving SIS using combinatorial techniques and

no bound on m.

Contents

Gaussian elimination

Asume $\mathbf{e} = \mathbf{0}$, we hence want to decide whether there is a solution $\mathbf{s}$ such that $\mathbf{c} = \mathbf{A} \cdot \mathbf{s}$. We may apply Gaussian elimination to the matrix:

$$[\mathbf{A} \mid \mathbf{c}] = \left(\begin{array}{cccc|c} \mathbf{a}_{11} & \mathbf{a}_{12} & \dots & \mathbf{a}_{1n} & c_1 \\ \mathbf{a}_{21} & \mathbf{a}_{22} & \dots & \mathbf{a}_{2n} & c_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{a}_{m1} & \mathbf{a}_{m2} & \dots & \mathbf{a}_{mn} & c_m \end{array}\right)$$

to recover

$$[\tilde{\mathbf{A}} \mid \tilde{\mathbf{c}}] = \left(\begin{array}{cccc|c} \mathbf{a}_{11} & \mathbf{a}_{12} & \dots & \mathbf{a}_{1n} & c_1 \\ 0 & \tilde{\mathbf{a}}_{22} & \dots & \tilde{\mathbf{a}}_{2n} & \tilde{c}_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & \tilde{\mathbf{a}}_{mn} & \tilde{c}_n \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & \tilde{c}_m \end{array}\right)$$

If and only if $\tilde{c}_{n+1}, \dots, \tilde{c}_m$ are all zero, the system is consistent.

Contents

BKW Algorithm I

The BKW algorithm was first proposed for the Learning Parity with Noise (LPN) problem which can be viewed as a special case of LWE.

Avrim Blum, Adam Kalai, and Hal Wasserman.
Noise-tolerant learning, the parity problem, and the statistical query model.
J. ACM, 50(4):506–519, 2003.

BKW Algorithm II

We revisit Gaussian elimination:

$$
\left(\begin{array}{c|c|c|ccc|c}
\mathbf{a}_{11} & \mathbf{a}_{12} & \mathbf{a}_{13} & \cdots & \mathbf{a}_{1n} & c_1 \\
\mathbf{a}_{21} & \mathbf{a}_{22} & \mathbf{a}_{23} & \cdots & \mathbf{a}_{2n} & c_2 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \\
\mathbf{a}_{m1} & \mathbf{a}_{m2} & \mathbf{a}_{m3} & \cdots & \mathbf{a}_{mn} & c_m
\end{array}\right)
$$

$$
= \left(\begin{array}{c|c|c|ccc|c}
\mathbf{a}_{11} & \mathbf{a}_{12} & \mathbf{a}_{13} & \cdots & \mathbf{a}_{1n} & \langle \mathbf{a}_1, \mathbf{s} \rangle + \mathbf{e}_1 \\
\mathbf{a}_{21} & \mathbf{a}_{22} & \mathbf{a}_{23} & \cdots & \mathbf{a}_{2n} & \langle \mathbf{a}_2, \mathbf{s} \rangle + \mathbf{e}_2 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \\
\mathbf{a}_{m1} & \mathbf{a}_{m2} & \mathbf{a}_{m3} & \cdots & \mathbf{a}_{mn} & \langle \mathbf{a}_m, \mathbf{s} \rangle + \mathbf{e}_m
\end{array}\right)
$$

$$
\Rightarrow \left(\begin{array}{c|c|c|ccc|c}
\mathbf{a}_{11} & \mathbf{a}_{12} & \mathbf{a}_{13} & \cdots & \mathbf{a}_{1n} & \langle \mathbf{a}_1, \mathbf{s} \rangle + \mathbf{e}_1 \\
0 & \tilde{\mathbf{a}}_{22} & \tilde{\mathbf{a}}_{23} & \cdots & \tilde{\mathbf{a}}_{2n} & \langle \tilde{\mathbf{a}}_2, \mathbf{s} \rangle + \mathbf{e}_2 - \frac{\mathbf{a}_{21}}{\mathbf{a}_{11}} \mathbf{e}_1 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \\
0 & \tilde{\mathbf{a}}_{m2} & \tilde{\mathbf{a}}_{m3} & \cdots & \tilde{\mathbf{a}}_{mn} & \langle \tilde{\mathbf{a}}_m, \mathbf{s} \rangle + \mathbf{e}_m - \frac{\mathbf{a}_{m1}}{\mathbf{a}_{11}} \mathbf{e}_1
\end{array}\right)
$$

BKW Algorithm III

- $\frac{\mathbf{a}_{i1}}{\mathbf{a}_{11}}$ is essentially a random element in $\mathbb{Z}_q$, hence $\tilde{c}_i \leftarrow_{\$} \mathcal{U}(\mathbb{Z}_q)$.
- Even if $\frac{\mathbf{a}_{i1}}{\mathbf{a}_{11}}$ is 1 the variance of the noise doubles at every level because of the addition.

Setting

Problem: additions and multiplications $\Rightarrow$ noise of $\tilde{c}$ values increases rapidly

Strategy: exploit that we have many rows: $m \gg n$.

BKW Algorithm IV

We considering $a \approx \log n$ 'blocks' of b elements each.

$$\left(\begin{array}{cc|ccc|c} \mathbf{a}_{11} & \mathbf{a}_{12} & \mathbf{a}_{13} & \cdots & \mathbf{a}_{1n} & c_0 \\ \mathbf{a}_{21} & \mathbf{a}_{22} & \mathbf{a}_{23} & \cdots & \mathbf{a}_{2n} & c_1 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \\ \mathbf{a}_{m1} & \mathbf{a}_{m2} & \mathbf{a}_{m3} & \cdots & \mathbf{a}_{mn} & c_m \end{array}\right)$$

BKW Algorithm V

For each block we build a table of all q^b possible values.

$$T = \left[\begin{array}{cc|ccc|c} 0 & 0 & \mathbf{a}_{13} & \cdots & \mathbf{a}_{1n} & c_0 \\ 0 & 1 & \mathbf{a}_{23} & \cdots & \mathbf{a}_{2n} & c_1 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \\ q & q & \mathbf{a}_{q^2 3} & \cdots & \mathbf{a}_{q^2 n} & c_{q^2} \end{array}\right]$$

BKW Algorithm VI

We use these tables to eliminate b entries in other rows.

$$
\left(\begin{array}{cc|ccc|c}
\mathbf{a}_{11} & \mathbf{a}_{12} & \mathbf{a}_{13} & \cdots & \mathbf{a}_{1n} & c_0 \\
\mathbf{a}_{21} & \mathbf{a}_{22} & \mathbf{a}_{23} & \cdots & \mathbf{a}_{2n} & c_1 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \\
\mathbf{a}_{m1} & \mathbf{a}_{m2} & \mathbf{a}_{m3} & \cdots & \mathbf{a}_{mn} & c_m
\end{array}\right)
$$

$$
+ \left[\begin{array}{cc|ccc|c}
0 & 0 & \mathbf{a}_{13} & \cdots & \mathbf{a}_{1n} & c_0 \\
0 & 1 & \mathbf{a}_{23} & \cdots & \mathbf{a}_{2n} & c_1 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \\
q & q & \mathbf{a}_{q^2 3} & \cdots & \mathbf{a}_{q^2 n} & c_{q^2}
\end{array}\right]
$$

$$
\Rightarrow \left(\begin{array}{cc|ccc|c}
\mathbf{a}_{11} & \mathbf{a}_{12} & \mathbf{a}_{13} & \cdots & \mathbf{a}_{1n} & c_0 \\
0 & 0 & \mathbf{a}_{23} & \cdots & \tilde{\mathbf{a}}_{2n} & \tilde{c}_1 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \\
0 & 0 & \tilde{\mathbf{a}}_{m3} & \cdots & \tilde{\mathbf{a}}_{mn} & \tilde{c}_m
\end{array}\right)
$$

BKW Algorithm VII

This gives a time complexity of

$$\approx (a^2 n) \cdot \frac{q^b}{2}$$

and a memory requirement of

$$\approx \frac{q^b}{2} \cdot a \cdot (n+1).$$

A detailed analysis of the algorithm for LWE is available as:

Martin R. Albrecht, Carlos Cid, Jean-Charles Faugère, Robert Fitzpatrick and Ludovic Perret
On the Complexity of the BKW Algorithm on LWE
ePrint Report 2012/636, 2012.
to appear in *Designs, Codes and Cryptography*.

Contents

The Setting

Assume $\mathbf{s} \leftarrow_{\$} \mathcal{U}(\mathbb{Z}_2^n)$, i.e. all entries in $\mathbf{s}$ are very small.

This is a common setting in cryptography for performance reasons and because this allows to realise some advanced schemes. In particular, a technique called 'modulus switching' can be used to improve the performance of homomorphic encryption schemes.

Zvika Brakerski and Vinod Vaikuntanathan.
Efficient fully homomorphic encryption from (standard) LWE.
In Rafail Ostrovsky, editor, *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011*, pages 97–106. IEEE, 2011.

Modulus Reduction I

Given a sample $(\mathbf{a}, c)$ where $c = \langle \mathbf{a}, \mathbf{s} \rangle + e$ and some $p < q$ we may consider

$$\left(\left\lfloor \frac{p}{q} \cdot \mathbf{a} \right\rceil, \left\lfloor \frac{p}{q} \cdot c \right\rceil \right)$$

with

$$\begin{aligned}
\left\lfloor \frac{p}{q} \cdot c \right\rceil &= \left\lfloor \left\langle \frac{p}{q} \cdot \mathbf{a}, \mathbf{s} \right\rangle + \frac{p}{q} \cdot e \right\rceil \\
&= \left\lfloor \left\langle \left\lfloor \frac{p}{q} \cdot \mathbf{a} \right\rceil, \mathbf{s} \right\rangle + \left\langle \frac{p}{q} \cdot \mathbf{a} - \left\lfloor \frac{p}{q} \cdot \mathbf{a} \right\rceil, \mathbf{s} \right\rangle + \frac{p}{q} \cdot e \right\rceil \\
&= \left\langle \left\lfloor \frac{p}{q} \cdot \mathbf{a} \right\rceil, \mathbf{s} \right\rangle + \left\langle \frac{p}{q} \cdot \mathbf{a} - \left\lfloor \frac{p}{q} \cdot \mathbf{a} \right\rceil, \mathbf{s} \right\rangle + \frac{p}{q} \cdot e \pm [0, 0.5] \\
&= \left\langle \left\lfloor \frac{p}{q} \cdot \mathbf{a} \right\rceil, \mathbf{s} \right\rangle + e''.
\end{aligned}$$

Modulus Reduction II

Example

$$
\begin{aligned}
p, q &= 10, 20 \\
\mathbf{a} &= (8, -2, 0, 4, 2, -7), \\
\mathbf{s} &= (0, 1, 0, 0, 1, 1), \\
\langle \mathbf{a}, \mathbf{s} \rangle &= -7, \\
c &= -6 \\
\mathbf{a}' = \left\lfloor \frac{p}{q} \cdot \mathbf{a} \right\rceil &= (4, -1, 0, 2, 1, -4) \\
\langle \mathbf{a}', \mathbf{s} \rangle &= -4, \\
\left\lfloor \frac{p}{q} \cdot c \right\rceil &= -4.
\end{aligned}
$$

Modulus Reduction III

Typically, we would choose

$$p \approx q \cdot \sqrt{n \cdot \mathrm{Var}(\mathcal{U}([-0.5, 0.5])) \cdot \sigma_s^2}/\sigma = q \cdot \sqrt{n/12}\sigma_s/\sigma$$

where σ_s is the standard deviation of elements in $\mathbf{s}$.

If $\mathbf{s}$ is small then e'' is small and we may compute with the smaller 'precision' p at the cost of a slight increase of the noise rate.

The complexity hence drops to

$$\approx (a^2 n) \cdot \frac{p^b}{2}$$

with a usually is unchanged.

Lazy Modulus Switching I

For simplicity assume $p = 2^\kappa$ and consider the LWE matrix

$$[\mathbf{A} \mid \mathbf{c}] = \begin{pmatrix} \mathbf{a}_{1,1} & \mathbf{a}_{1,2} & \cdots & \mathbf{a}_{1,n} & c_1 \\ \mathbf{a}_{2,1} & \mathbf{a}_{2,2} & \cdots & \mathbf{a}_{2,n} & c_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{a}_{m,1} & \mathbf{a}_{m,2} & \cdots & \mathbf{a}_{m,n} & c_m \end{pmatrix}$$

as

$$[\mathbf{A} \mid \mathbf{c}] = \begin{pmatrix} \mathbf{a}^h_{1,1} & \mathbf{a}^l_{1,1} & \mathbf{a}^h_{1,2} & \mathbf{a}^l_{1,2} & \cdots & \mathbf{a}^h_{1,n} & \mathbf{a}^l_{1,n} & c_1 \\ \mathbf{a}^h_{2,1} & \mathbf{a}^l_{2,1} & \mathbf{a}^h_{2,2} & \mathbf{a}^l_{2,2} & \cdots & \mathbf{a}^h_{2,n} & \mathbf{a}^l_{2,n} & c_2 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \mathbf{a}^h_{m,1} & \mathbf{a}^l_{m,1} & \mathbf{a}^h_{m,2} & \mathbf{a}^l_{m,2} & \cdots & \mathbf{a}^h_{m,n} & \mathbf{a}^l_{m,n} & c_m \end{pmatrix}$$

where $\mathbf{a}^h_{i,j}$ and $\mathbf{a}^l_{i,j}$ denote high and low order bits:

- $\mathbf{a}^h_{i,j}$ corresponds to $\lfloor p/q \cdot \mathbf{a}_{i,j} \rceil$ and
- $\mathbf{a}^l_{i,j}$ corresponds to $\lfloor p/q \cdot \mathbf{a}_{i,j} \rceil - p/q \cdot \mathbf{a}_{i,j}$, the rounding error.

Lazy Modulus Switching II

In order to clear the most significant bits in every component of the $\mathbf{a}_i$, we run the BKW algorithm on the matrix $[\mathbf{A} \mid \mathbf{c}]$ but only consider

$$[\mathbf{A}, \mathbf{c}]^h := \begin{pmatrix} \mathbf{a}_{1,1}^h & \mathbf{a}_{1,2}^h & \cdots & \mathbf{a}_{1,n}^h & c_1 \\ \mathbf{a}_{2,1}^h & \mathbf{a}_{2,2}^h & \cdots & \mathbf{a}_{2,n}^h & c_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{a}_{m,1}^h & \mathbf{a}_{m,2}^h & \cdots & \mathbf{a}_{m,n}^h & c_m \end{pmatrix}.$$

when searching for collisions.

We only manage elimination tables for the most significant κ bits.
All arithmetic is performed in $\mathbb{Z}_q$ but collisions are searched for in $\mathbb{Z}_p$.

Lazy Modulus Switching III

- We do not apply modulus reduction in one shot, but only when needed
- As a consequence rounding errors accumulate not as fast: they only start to accumulate when we branch on a component.

We may reduce p by a factor of $\sqrt{a/2}$.

This may translate to huge gains the complexity of BKW is $\approx p^b$ where typically $b \approx n/\log n$.

Stunting Growth I

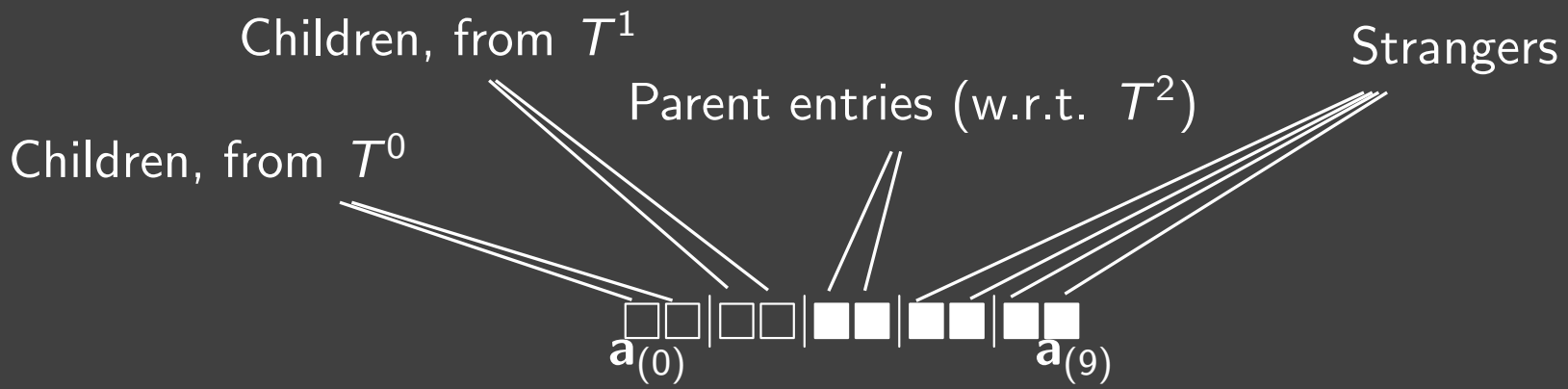

Figure : Children, parents and strangers.

Stunting Growth II

Assume $b = 1$ and $a \geq 3$, for the outputs $(\tilde{\mathbf{a}}_i, \tilde{c}_i)$ where the first three components are reduced have:

$$\begin{aligned} \tilde{\mathbf{a}}_i &= \mathbf{a}_i \text{ from } L^{(n)}_{\mathbf{s},\chi} \\ &+ \tilde{\mathbf{a}}_0 \text{ with } \tilde{\mathbf{a}}_0 \text{ from } T^0 \\ &+ \tilde{\mathbf{a}}_1 \text{ with } \tilde{\mathbf{a}}_1 \text{ from } T^1 \\ &+ \tilde{\mathbf{a}}_2 \text{ with } \tilde{\mathbf{a}}_2 \text{ from } T^2 \end{aligned}$$

Considering component $\tilde{\mathbf{a}}_{i,(0)}$ we have that

- $\mathbf{a}_{i,(0)}$ is uniform in $\mathbb{Z}_q$,
- $\tilde{\mathbf{a}}_{0,(0)}$ reduces this to something of size $r = \log_2 q - \log_2 p$
- $\tilde{\mathbf{a}}_{1,(0)}$ has size $\log_2 q - \log_2 p$
- $\tilde{\mathbf{a}}_{2,(0)}$ has size $\approx \log_2 q - \log_2 p + 1$, and **depends** on entries on T^1.

Stunting Growth III

We sample many candidates for $\tilde{\mathbf{a}}_2$ to find one where $\tilde{\mathbf{a}}_{2,(0)}$ is particularly small.

This is easier than for $\tilde{\mathbf{a}}_3$ but influences $\tilde{\mathbf{a}}_3$.

Stunting Growth IV

Assumption

Let the vectors $\mathbf{x}_i \in \mathbb{Z}_q^\tau$ *be sampled from some distribution* $\mathcal{D}$ *such that* $\sigma^2 = \mathrm{Var}(\mathbf{x}_{i,(j)})$ *where* $\mathcal{D}$ *is any distribution on (sub-)vectors observable in our algorithm. Let* $\mathbf{y} = \min_{abs}(\mathbf{x}_0, \ldots, \mathbf{x}_{n-1})$ *where* $\min_{abs}$ *picks that vector* $\mathbf{x}_{min}$ *with* $\sum_{j=0}^{b\cdot\ell-1} |\mathbf{x}_{min,(j)}|$ *minimal. The standard deviation* $\sigma_n = \sqrt{\mathrm{Var}(\mathbf{y}_{(j)})}$ *of components in* $\mathbf{y}$ *satisfies*

$$\sigma/\sigma_n \geq c_\tau \sqrt[\tau]{n} + (1 - c_\tau)$$

with

$$c_\tau = 0.20151418166952917\sqrt{\tau} + 0.32362108131969386 \approx \frac{1}{5}\sqrt{\tau} + \frac{1}{3}.$$

Martin R. Albrecht, Jean-Charles Faugère, Robert Fitzpatrick and Ludovic Perret
Lazy Modulus Switching for the BKW Algorithm on LWE
in preparation, 2013.

Results

	BKW		+ Mod. Switch	
n	$\log \mathbb{Z}_2$	log mem	$\log \mathbb{Z}_2$	log mem
32	40.0	26.2	39.4	25.5
64	55.9	48.8	52.5	46.0
128	97.6	90.0	89.6	81.2
256	182.1	174.2	164.0	156.7
512	361.0	352.8	305.6	297.9
1024	705.5	697.0	580.2	572.2
	This Work (1)		This Work (2)	
n	$\log \mathbb{Z}_2$	log mem	$\log \mathbb{Z}_2$	log mem
32	40.0	26.1	40.0	26.1
64	49.2	42.1	47.6	32.0
128	78.2	70.8	74.2	46.3
256	142.7	134.9	132.5	67.1
512	251.2	243.1	241.8	180.0
1024	494.8	486.5	485.0	407.5

Table : Cost for solving Decision-LWE with advantage ≈ 1 for BKW and BKZ variants where q and σ are chosen as in Regev's scheme and $\mathbf{s} \leftarrow_{\$} \mathcal{U}(\mathbb{Z}_2^n)$ "$\log \mathbb{Z}_2$" gives the number of "bit operations" and "log mem" the memory requirement of $\mathbb{Z}_q$ elements. All logarithms are base 2.

Fin

Questions?