Multilinear Maps From Ideal Lattices

Sanjam Garg (IBM)
Joint work with
Craig Gentry (IBM) and Shai Halevi (IBM)
Outline

• Bilinear Maps: Recall and Applications
  • Motivating Multilinear maps

• Our Results

• Definitions of Multi-linear Maps
  • Classical Notion
  • Our Notion

• Our Construction
  • Security
Cryptographic Bilinear Maps
(Weil and Tate Pairings)

Recalling Bilinear Maps and its Applications: Motivating Multilinear Maps
Cryptographic Bilinear Maps

- Bilinear maps are extremely useful in cryptography
  - lots of applications

- As the name suggests allow pairing two things together
Bilinear Maps – Definitions

• Cryptographic bilinear map
  • Groups $G_1$ and $G_2$ of order $p$ with generators $g_1, g_2 = e(g_1, g_1)$ and a bilinear map $e : G_1 \times G_1 \rightarrow G_2$ such that

$$\forall a, b \in \mathbb{Z}_p, \quad e(g_1^a, g_1^b) = g_2^{ab}$$

• Instantiation: Weil or Tate pairings over elliptic curves.

CDH is hard
Given $g_1^a, g_1^b$ hard to get $g_1^{ab}$

DDH is easy
Given $g_1^a, g_1^b, T$

$$T \equiv g_1^{ab}$$

$$e(g_1^a, g_1^b) = e(g_1, T)$$
Bilinear Maps: "Hard" Problem

• Bilinear Diffie-Hellman: Given

\[ g_1, g_1^a, g_1^b, g_1^c \in G_1 \text{ hard to distinguish } e(g_1, g_1^{abc}) = g_2^{abc} \text{ from Random} \]
Non-Interactive Key Agreement \([\text{DH76]}\)

Application 1

- Easy Application: \textbf{Tri-partite key} agreement [Joux00]:
  - Alice, Bob, Carol generate \(a, b, c\) and broadcast \(g_1^a, g_1^b, g_1^c\).
  - They each separately compute the key \(K = e(g_1^a, g_1^b)^{abc}\).

- What if we have more than \textbf{3-parties}? [BS03]
Outline

• Bilinear Maps: Recall and Applications
  • Motivating Multilinear maps

• Our Results

• Definitions of Multi-linear Maps
  • Classical Notion
  • Our Notion

• Our Construction
  • Security
Our Results

• **Candidate approximate** constructions of multi-linear maps

• Lots of Applications:
  • Witness Encryption
  • Indistinguishability Obfuscation
Application 2

Witness Encryption [GGSW13]

[TW87, Rudich89, IOS97, IS91, KMV07, CS02, CCKV08, GOVW12 ...]

Soundness:
Statement is false $\implies$ Semantic Security
Application 3

Indistinguishability Obfuscation

[GGHRSW13]

[Barak et al...]

\[ O(C) \]

\[ C \]

Obfuscator

\[ O(C) \]

Security: Can’t tell if \( C = C_1 \) or \( C_2 \)
As long as \( \forall x \, C_1(x) = C_2(x) \) and \( |C_1| = |C_2| \)
Outline

• Bilinear Maps: Recall and Applications
  • Motivating Multilinear maps
• Our Results
• Definitions of Multi-linear Maps
  • Classical Notion
  • Our Notion
• Our Construction
  • Security
Cryptographic Multi-linear Maps

Definitions: Classical notion and our Approximate variant
Multilinear Maps: Classical Notion

- Cryptographic $n$-multilinear map (for groups)
  - Groups $G_1, \ldots, G_n$ of order $p$ with generators $g_1, \ldots, g_n$
  - Family of maps:
    \[ e_{i,k} : G_i \times G_k \rightarrow G_{i+k} \text{ for } i + k \leq n, \text{ where} \]
    \[ e_{i,k}(g_i^a, g_k^b) = g_{i+k}^{ab} \quad \forall a, b \in \mathbb{Z}_p. \]
- And at least the "discrete log" problems in each $G_i$ is "hard".
  - And hopefully the generalization of Bilinear DH
Getting to our Notion

- Our visualization of (traditional) Bilinear Maps
- Step by step I will make changes to get our notion of Bilinear Maps
- At each step provide Extension to Multi-linear Maps
Bilinear Maps: Our visualization

\[
\begin{array}{ccc}
Z_p & G_1 & G_2 \\
1 & g_1^1 & g_2^1 \\
2 & g_1^2 & g_2^2 \\
\vdots & \vdots & \vdots \\
p & g_1^p & g_2^p \\
\end{array}
\]
Bilinear Maps: Our visualization

Sampling

\[ Z_p \]

\[ \begin{array}{ccc}
G_1 & & G_2 \\
1 & g^1_1 & \vdots \ & g^1_2 \\
2 & g^2_1 & \vdots \ & g^2_2 \\
\vdots & \vdots & \ddots & \vdots \\
p & g^p_1 & \vdots & g^p_2 \\
\end{array} \]

It was easy to sample uniformly from \( Z_p \).
Bilinear Maps: Our visualization
Equality Checking

<table>
<thead>
<tr>
<th>$\mathbb{Z}_p$</th>
<th>$G_1$</th>
<th>$G_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$g_1^1$</td>
<td>$g_2^1$</td>
</tr>
<tr>
<td>2</td>
<td>$g_1^2$</td>
<td>$g_2^2$</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
<td>:</td>
</tr>
<tr>
<td>$p$</td>
<td>$g_1^p$</td>
<td>$g_2^p$</td>
</tr>
</tbody>
</table>

Trivial to check if two terms are the same.
Bilinear Maps: Our visualization
Addition

\[ Z_p \]
\[ 1 \]
\[ 2 \]
\[ \vdots \]
\[ p \]
\[ g_1^3 \]
\[ g_1^1 \]
\[ g_1^2 \]
\[ g_1^p \]

\[ G_1 \]
\[ g_1^1 \]
\[ g_1^2 \]
\[ g_1^p \]

\[ G_2 \]
\[ g_2^1 \]
\[ g_2^2 \]
\[ g_2^p \]
Bilinear Maps: Our visualization
Multiplication

\[
\begin{array}{ccc}
Z_p & G_1 & G_2 \\
1 & g_1^1 & g_2^1 \\
2 & g_1^2 & g_2^2 \\
\vdots & \vdots & \vdots \\
p & g_1^p & g_2^p \\
\end{array}
\]
Bilinear Maps: Sets
(Our Notion)

\[
\begin{align*}
Z_p & \quad G_1 & \quad G_2 \\
1 & S_0^1 & g_1^1 & S_1^1 & g_2^1 & S_2^1 \\
2 & S_0^2 & g_1^2 & S_1^2 & g_2^2 & S_2^2 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
p & S_0^p & g_1^p & S_1^p & g_2^p & S_2^p \\
\end{align*}
\]

Level-0 encodings
Multilinear Maps: Our Notion

- Finite ring $R$ and sets $S_i \forall i \in [n]$: "level-$i$ encodings"
- Each set $S_i$ is partitioned into $S_i^a$ for each $a \in R$: "level-$i$ encodings of $a$".
It was easy to sample uniformly from $Z_p$.

I should be efficient to sample $\alpha \leftarrow S_0$ such that $\alpha \in S_0^a$ for a uniform $\alpha$. It may not be uniform in $S_0$ or $S_0^a$. 

It was easy to sample uniformly from $Z_p$. 
Multilinear Maps: Our Notion

- Finite ring $R$ and sets $S_i \forall i \in [n]$: “level-$i$ encodings”
- Each set $S_i$ is partitioned into $S_i^a$ for each $a \in R$: “level-$i$ encodings of $a$”.
- **Sampling**: Output $\alpha$ such that $\alpha \in S_0^a$ for a uniform $\alpha$
### Bilinear Maps: Equality Checking
(Our Notion)

<table>
<thead>
<tr>
<th>$Z_p$</th>
<th>$G_1$</th>
<th>$G_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$s_0^1$</td>
<td>$g_1^1$</td>
</tr>
<tr>
<td>2</td>
<td>$s_0^2$</td>
<td>$g_1^2$</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
<td>:</td>
</tr>
<tr>
<td>$p$</td>
<td>$s_0^p$</td>
<td>$g_1^p$</td>
</tr>
</tbody>
</table>

Check if two values come from the same set.

It was trivial to check if two terms are the same.
Multilinear Maps: Our Notion

- Finite ring $R$ and sets $S_i \forall i \in [n]$: “level-i encodings”
- Each set $S_i$ is partitioned into $S_i^a$ for each $a \in R$: “level-i encodings of $a$”.
- **Sampling**: Output $\alpha$ such that $\alpha \in S_0^a$ for a random $a$
- **Equality testing($\alpha, \beta, i$)**: Output 1 iff $\exists a$ such that $\alpha, \beta \in S_i^a$
Bilinear Maps: Addition
(Our Notion)
Multilinear Maps: Our Notion

- Finite ring $R$ and sets $S_i \forall i \in [n]$: \textquote{``level-i encodings''}
- Each set $S_i$ is partitioned into $S_i^a$ for each $a \in R$: \textquote{``level-i encodings of $a$''}.
- Sampling: Output $\alpha$ such that $\alpha \in S_0^a$ for a random $a$
- Equality testing($\alpha, \beta, i$): Output 1 iff $\exists a$ such that $\alpha, \beta \in S_i^a$
- Addition/Subtraction: There are ops $+$ and $-$ such that:
  - $\forall i \in [n], a, b \in R, \alpha \in S_i^a, \beta \in S_i^b$:
  - We have $\alpha + \beta \in S_i^{a+b}$ and $\alpha - \beta \in S_i^{a-b}$. 
Bilinear Maps: Multiplication
(Our Notion)

\[
\begin{align*}
Z_p & \quad G_1 & \quad G_2 \\
1 & S_0^1 & S_1^1 & S_2^1 \\
2 & S_0^2 & S_1^2 & S_2^2 \\
\vdots & \vdots & \vdots & \vdots \\
p & S_0^p & S_1^p & S_2^p \\
S_0 & S_1 & S_2
\end{align*}
\]
Multilinear Maps: Our Notion

- Finite ring $R$ and sets $S_i \forall i \in [n]$: “level-$i$ encodings”
- Each set $S_i$ is partitioned into $S_i^a$ for each $a \in R$: “level-$i$ encodings of $a$”.
- Sampling: Output $\alpha$ such that $\alpha \in S_0^a$ for a random $a$
- Equality testing($\alpha$, $\beta$, $i$): Output 1 iff $\exists a$ such that $\alpha, \beta \in S_i^a$
- Addition/Subtraction: There are ops $+$ and $-$ such that:
  - Multiplication: There is an op $\times$ such that:
    - $\forall i, k$ such that $i + k \leq n$, $\forall a, b \in R, \alpha \in S_i^a, \beta \in S_k^b$:
    - We have $\alpha \times \beta \in S_{i+k}^{ab}$. 
Bilinear Maps: **Noisy**
(Our Notion)

All operations are required to work as long as "noise" level remains small.
Multilinear Maps: Our Notion

- **Discrete Log**: Given level-$j$ encoding of $a$, hard to compute level-$(j-1)$ encoding of $a$.

- **n-Multilinear DDH**: Given level-1 encodings of $1, a_1, \ldots, a_{n+1}$ and a level-$n$ encoding $T$ distinguish whether $T$ encodes $a_1 \cdots a_{n+1}$ or not.
Outline

• Bilinear Maps: Recall and Applications
  • Motivating Multilinear maps
• Our Results
• Definitions of Multi-linear Maps
  • Classical Notion
  • Our Notion
• Our Construction
  • Security
``Noisy” Multilinear Maps

(Kind of like NTRU-Based FHE, but with Equality Testing)
Our Construction

- We work in polynomial ring $R = \mathbb{Z}[x]/f(x)$
  - E.g., $f(x) = x^n + 1$ ($n$ is a power of two)
  - Also use $R_q = \mathbb{R}/qR = \mathbb{Z}[x]/(f(x), q)$

- Public parameters hide a small $g \in R_q$ and a random (large) $z \in R_q$
  - $g$ defines a principal ideal $I = (g)$ over $R$
  - The ``scalars'' that we encode are cosets of $I$ (i.e., elements in the quotient ring $R/I$)
    - e.g., if $|R/I| = p$ is a prime, then we can represent these cosets using the integers $1, 2, \ldots, p$
Our Construction

• $R = \mathbb{Z}[x]/f(x)$ and $R_q = R/qR$

• Small $g \in R_q$ defines a principal ideal $I = (g)$ over $R$

• $A$ random (large) $z \in R_q$ and $c$ should have small coefficients
Our Construction

- \( R = \mathbb{Z}[x]/f(x) \) and \( R_q = R/qR \)

- Small \( g \in R_q \) defines a principal ideal \( I = (g) \) over \( R \)

\[ \begin{align*}
S_0^1 & \quad 1 + I \\
S_0^2 & \quad 2 + I \\
\vdots & \quad \vdots
\end{align*} \]

\[ \begin{align*}
S_1^1 & \quad c \\
S_1^2 & \quad \left[ \frac{c}{z} \right]_q \\
S_2^1 & \quad \left[ \frac{c}{z^2} \right]_q \\
S_2^2 & \quad \vdots
\end{align*} \]

+ and \( \times \)

**Addition**

If \( c \in s + I, d \in t + I \), are both short then,

\[ \left[ \frac{c}{z} + \frac{d}{z} \right]_q \text{ has the form } \left[ \frac{c+d}{z} \right]_q, \]

where \( c + d \) is still short and \( c + d \in s + t + I \)

- A random (large) \( z \in R_q \) \( c \) should have small coefficients
Our Construction

- \( R = \mathbb{Z}[x]/f(x) \) and \( R_q = R/qR \)
- Small \( g \in R_q \) defines a principal ideal \( I = (g) \) over \( R \)

\[
\begin{align*}
S^1_0 & \quad 1 + I \quad S^1_1 \\
S^2_0 & \quad 2 + I \\
\vdots & \quad \vdots \\
S^1_0 & \quad c \\
S^2_1 & \quad \left[ \frac{c}{z} \right]_q \\
S^2_0 & \quad \left[ \frac{c}{z^2} \right]_q
\end{align*}
\]

If \( c \in s + I, d \in t + I \), are both short then,
\[
\left[ \frac{c \times d}{z} \right]_q \text{ has the form } \left[ \frac{c \times d}{z^2} \right]_q,
\]
where \( c \times d \) is still short and \( c \times d \in s \cdot t + I \)

- A random (large) \( z \in R_q \) \( c \) should have small coefficients
Our Construction (in general)

• In general, “level-k encoding” of a coset \( s + I \) has the form \( \left[ \frac{c}{z^k} \right]_q \) for a short \( c \in s + I \)

• **Addition**: Add encodings \( u_i = \left[ \frac{c_i}{z^{j_i}} \right]_q \)
  • as long as \( |\sum_i c_i| \ll q \)

• **Multi-linear**: Multiply encodings \( u_i = \left[ \frac{c_i}{z^{j_i}} \right]_q \)
  • to get an encoding of the product at level \( \sum_i j_i \)
  • as long as \( |\prod_i c_i| \ll q \)

• “Somewhat homomorphic” encoding

Sampling and equality check?
Sampling

- **Sampling**: If \( c \leftarrow \text{DiscreteGaussian}(Z^n) \) (wider than smoothing parameter [MR05] of \( g \) but still smaller than \( q \)), then \( c \) encodes a random coset.
  - Why should this work?
  - Recall \( I = (g) \) -- vector with tiny coefficients
Encoding this random coset

- Publish an encoding of 1:
  - $y = [a/z]_q$

- **Sampling**: If $c \leftarrow \text{DiscreteGaussian}(Z^n)$ (wide enough), then $c$ encodes a random coset.
  - Don’t know how to encode specific elements

- Given this short $c$, set $u = [c \cdot y]_q$
  - $u$ is a valid level-1 encoding of the coset $c + I$

- Translating from level $i$ to $i + 1$: $u_{i+1} = [u_i \cdot y]_q$
Equality Checking

• Do $u, u'$ encode the same coset?
  • Suffices to check $-\left[u - u'\right]_q$ encodes 0.
• Publish a (level-$k$) zero-testing param
  $$v_k = \left[\frac{hz^k}{g}\right]_q$$
  • $h$ is "somewhat short" (e.g. of size $\sqrt{q}$)
• To test, if $u = \left[\frac{c}{z^k}\right]_q$ encodes 0, compute
  • $w = \left[u \cdot v_k\right]_q = \left[\frac{c}{z^k} \cdot \frac{hz^k}{g}\right]_q = \left[\frac{ch}{g}\right]_q$
  • Which is small if $c \in I$ (or, $c = c'g$)
Re-randomization

- Compute $c_{st} = c_s c_t$
- And encode $u_s = [c_s y]_q, u_t = [c_t y]_q, u_{st} = [c_{st} y]_q$
  - But then $u_{st} = \frac{u_s u_t}{y}$
- We need to re-randomize the encoding, to break these simple algebraic relations
Re-randomization

This re-randomization gets us statistically close to the actual distribution [AGHS12].

Need to re-randomize this as well.
The Complete Encoding Scheme

- Parameters:
  \[ y = \left[ \frac{a}{z} \right]_q, \quad \{x_i = \left[ \frac{b_i}{z} \right]_q \}_i, \quad \text{and} \quad v_k = \left[ \frac{hz^k}{g} \right]_q \]

- Encode a random element:
  - Sample \( c \) and set \( u = [cy + \sum_i \rho_i x_i]_q \)
  - \( \rho_i \leftarrow \text{DiscreteGaussian}_s(Z) \)

- Re-randomize \( u \) (at level 1):
  - \( u' = [u + \sum_i \rho_i x_i]_q \)

- Zero Test:
  - Map to level \( k \) (by multiplying by \( y^j \) for appropriate \( j \))
  - Check if \( [u \cdot v_k]_q \) is small
Variants

- Asymmetric variants (many $z_i$’s), XDH analog
  \[ y_i = \left[ \frac{a_i}{z_i} \right]_q, \quad \left\{ x_{i,j} = \left[ \frac{b_{i,j}}{z_i} \right]_q \right\}_{i,j}, \quad \nu_k = \left[ \frac{h \prod_i z_i}{g} \right]_q \]

- Partially symmetric and partially asymmetric
Security: Cryptanalysis
Assumptions

\[ y_0 = \left[ \frac{a_0}{z} \right]_q, \ldots, y_k = \left[ \frac{a_k}{z} \right]_q \text{ and } v_k = \left[ \frac{hz^k}{g} \right]_q \]

- **Goal:** Distinguish
  - \( \left[ \prod a_i \right]_q \) from \( \left[ \frac{r}{z^k} \right]_q \)

- **Easy**
  - \( \left\{ x_i = \left[ \frac{b_i}{z} \right]_q \right\}_i \)
    - General computation and not just multilinear

- **Difficult**
  - \( y_0 = \left[ \frac{a_0}{z_1} \right]_q, \ldots, y_k = \left[ \frac{a_k}{z_k} \right]_q \text{ and } v_k = \left[ \frac{h \prod z_i}{g} \right]_q \)
Attacks

\[ y = \left[ \frac{a}{z} \right]_q, \left\{ x_i = \left[ \frac{b_i}{z} \right]_q \right\}_i, \text{ and } v_k = \left[ \frac{hz^k}{g} \right]_q \]

- **Goal:** To find \( z \) or \( g \)
- **Covering the basics (Not ``Trivially’’ broken)**
  - Adversary that only (iteratively) adds, subtracts, multiplies, or divides pairs of elements that it has already computed cannot break the scheme
  - Similar in spirit to Generic Group model
- **Without the \( v_k \) - essentially the NTRU problem**
Some attacks

\[ y = \left[\frac{a}{z}\right]_q, \left\{ x_i = \left[\frac{b_i}{z}\right]_q \right\}_i, \text{ and } v_k = \left[\frac{hz^k}{g}\right]_q \]

- **Goal:** To find \( z \) or \( g \)
- **Can easily find ideal for** \( \langle h \rangle, \langle h \cdot g \rangle \) **and** \( \langle g \rangle \)

- **Can not hope to hide** \( I = \langle g \rangle \) **itself**
  - But not small
  - This is the basis for conjectured hardness
Summary

• Presented "noisy" cryptographic multilinear map.
• Construction is similar to NTRU-based homomorphic encryption, but with an equality-testing parameter.
• Security is based on somewhat stronger computational assumptions than NTRU.
• But more cryptanalysis needs to be done!
Thank You! Questions?