Practical Bootstrapping in Quasilinear Time

Jacob Alperin-Sheriff   Chris Peikert

School of Computer Science
Georgia Tech

UC San Diego
29 April 2013

Fully Homomorphic Encryption [RAD'78,Gen'09]

- FHE lets you do this:

$$\boxed{\mu} \longrightarrow \boxed{\mathsf{Eval}\left(f\ ,\ \boxed{\mu}\right)} \longrightarrow \boxed{f(\mu)}$$

where $|f(\mu)|$ and decryption time don't depend on $|f|$.

A cryptographic "holy grail" with tons of applications.

Fully Homomorphic Encryption [RAD'78,Gen'09]

- FHE lets you do this:

$$\boxed{\mu} \longrightarrow \boxed{\mathsf{Eval}\left(f\ ,\ \boxed{\mu}\right)} \longrightarrow \boxed{f(\mu)}$$

  where $|f(\mu)|$ and decryption time don't depend on $|f|$.

  A cryptographic "holy grail" with tons of applications.

- Naturally occurring schemes are "somewhat homomorphic" (SHE): they can only evaluate functions of an *a priori* bounded depth.

$$\boxed{\mu} \rightarrow \boxed{\mathsf{Eval}\left(f, \boxed{\mu}\right)} \rightarrow \boxed{f(\mu)} \rightarrow \boxed{\mathsf{Eval}\left(g, \boxed{f(\mu)}\right)} \rightarrow \boxed{g(f(\mu))}$$

Bootstrapping: SHE $\to$ FHE [Gen'09]

- Homomorphically evaluates the SHE decryption function to "refresh" a ciphertext $\boxed{\mu}$, allowing further homomorphic operations.

$$\boxed{sk} \longrightarrow \boxed{\text{Eval}\left(\ f(x) = \text{Dec}_x(\boxed{\mu})\ ,\ \boxed{sk}\ \right)} \longrightarrow \boxed{\mu}$$

Bootstrapping: SHE $\rightarrow$ FHE [Gen'09]

- Homomorphically evaluates the SHE decryption function to "refresh" a ciphertext $\boxed{\mu}$, allowing further homomorphic operations.

$$\boxed{sk} \longrightarrow \boxed{\mathsf{Eval}\left(f(x) = \mathsf{Dec}_x(\boxed{\mu}) \ , \ \boxed{sk} \right)} \longrightarrow \boxed{\mu}$$

  - ⋆ The only known way of obtaining unbounded FHE.
  - ⋆ Goal: Efficiency! Minimize depth d and size s of decryption "circuit."
  - ⋆ Best SHEs [BGV'12] can evaluate in time $\tilde{O}(d \cdot s \cdot \lambda)$.

Bootstrapping: SHE $\to$ FHE [Gen'09]

- Homomorphically evaluates the SHE decryption function to "refresh" a ciphertext $\boxed{\mu}$, allowing further homomorphic operations.

$$\boxed{sk} \longrightarrow \boxed{\mathsf{Eval}\left(f(x) = \mathsf{Dec}_x(\boxed{\mu})\ ,\ \boxed{sk} \right)} \longrightarrow \boxed{\mu}$$

  - ⋆ The only known way of obtaining unbounded FHE.
  - ⋆ Goal: Efficiency! Minimize depth d and size s of decryption "circuit."
  - ⋆ Best SHEs [BGV'12] can evaluate in time $\tilde{O}(d \cdot s \cdot \lambda)$.

- Intensive study, many techniques [G'09,GH'11a,GH'11b,GHS'12b], but still very inefficient – the main bottleneck in FHE, by far.

Bootstrapping: SHE → FHE [Gen'09]

- Homomorphically evaluates the SHE decryption function to "refresh" a ciphertext $\boxed{\mu}$, allowing further homomorphic operations.

$$\boxed{sk} \longrightarrow \boxed{\mathsf{Eval}\left(f(x) = \mathsf{Dec}_x(\boxed{\mu}) \;,\; \boxed{sk} \right)} \longrightarrow \boxed{\mu}$$

  - ⋆ The only known way of obtaining unbounded FHE.
  - ⋆ Goal: Efficiency! Minimize depth d and size s of decryption "circuit."
  - ⋆ Best SHEs [BGV'12] can evaluate in time $\tilde{O}(d \cdot s \cdot \lambda)$.

- Intensive study, many techniques [G'09,GH'11a,GH'11b,GHS'12b], but still very inefficient – the main bottleneck in FHE, by far.

- The asymptotically most efficient methods on "packed" ciphertexts [GHS'12a,GHS'12b] are very complex, and appear practically worse than asymptotically slower methods.

Milestones in Bootstrapping

[Gen'09]: $\tilde{O}(\lambda^4)$ runtime

Milestones in Bootstrapping

[Gen'09]: $\tilde{O}(\lambda^4)$ runtime

[BGV'12]: $\tilde{O}(\lambda^2)$ runtime, or $\tilde{O}(\lambda)$ amortized over λ ciphertexts

Milestones in Bootstrapping

[Gen'09]: $\tilde{O}(\lambda^4)$ runtime

[BGV'12]: $\tilde{O}(\lambda^2)$ runtime, or $\tilde{O}(\lambda)$ amortized over λ ciphertexts

Mainly via improved SHE homomorphic capacity.

Amortized method requires "exotic" plaintext rings, emulating $\mathbb{Z}_2$ arithmetic in $\mathbb{Z}_p$.

Milestones in Bootstrapping

[Gen'09]: $\tilde{O}(\lambda^4)$ runtime

[BGV'12]: $\tilde{O}(\lambda^2)$ runtime, or $\tilde{O}(\lambda)$ amortized over λ ciphertexts

Mainly via improved SHE homomorphic capacity.

Amortized method requires "exotic" plaintext rings, emulating $\mathbb{Z}_2$ arithmetic in $\mathbb{Z}_p$.

[GHS'12b]: $\tilde{O}(\lambda)$ runtime, for "packed" plaintexts. Declare victory?

Milestones in Bootstrapping

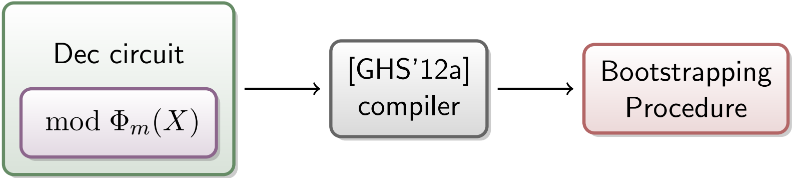
[Gen'09]: $\tilde{O}(\lambda^4)$ runtime

[BGV'12]: $\tilde{O}(\lambda^2)$ runtime, or $\tilde{O}(\lambda)$ amortized over λ ciphertexts

Mainly via improved SHE homomorphic capacity.

Amortized method requires "exotic" plaintext rings, emulating $\mathbb{Z}_2$ arithmetic in $\mathbb{Z}_p$.

[GHS'12b]: $\tilde{O}(\lambda)$ runtime, for "packed" plaintexts. Declare victory?



Milestones in Bootstrapping

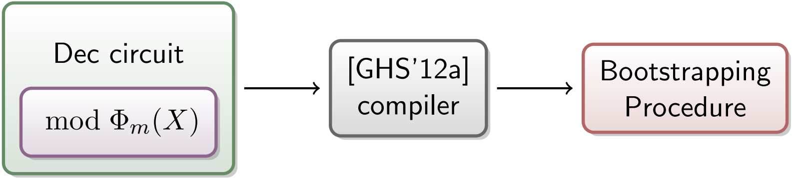
[Gen'09]: $\tilde{O}(\lambda^4)$ runtime

[BGV'12]: $\tilde{O}(\lambda^2)$ runtime, or $\tilde{O}(\lambda)$ amortized over λ ciphertexts

Mainly via improved SHE homomorphic capacity.

Amortized method requires "exotic" plaintext rings, emulating $\mathbb{Z}_2$ arithmetic in $\mathbb{Z}_p$.

[GHS'12b]: $\tilde{O}(\lambda)$ runtime, for "packed" plaintexts. Declare victory?



✗ Log-depth mod-$\Phi_m(X)$ circuit is complex, w/large hidden constants.

Milestones in Bootstrapping

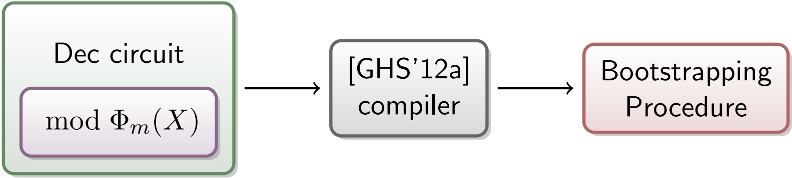
[Gen'09]: $\tilde{O}(\lambda^4)$ runtime

[BGV'12]: $\tilde{O}(\lambda^2)$ runtime, or $\tilde{O}(\lambda)$ amortized over λ ciphertexts

Mainly via improved SHE homomorphic capacity.

Amortized method requires "exotic" plaintext rings, emulating $\mathbb{Z}_2$ arithmetic in $\mathbb{Z}_p$.

[GHS'12b]: $\tilde{O}(\lambda)$ runtime, for "packed" plaintexts. Declare victory?



✗ Log-depth mod-$\Phi_m(X)$ circuit is complex, w/large hidden constants.

✗✗ [GHS'12a] compiler is very complex, w/large polylog overhead factor.

Our Results

Practical bootstrapping algorithms with quasi-linear $\tilde{O}(\lambda)$ runtimes:

Our Results

Practical bootstrapping algorithms with quasi-linear $\tilde{O}(\lambda)$ runtimes:

1. For "unpacked" (single-bit) plaintexts:
   - ✔ Extremely simple!
   - ✔ Uses only power-of-2 cyclotomic rings (fast, easy to implement).

Our Results

Practical bootstrapping algorithms with quasi-linear $\tilde{O}(\lambda)$ runtimes:

1. For "unpacked" (single-bit) plaintexts:
   - ✔ Extremely simple!
   - ✔ Uses only power-of-2 cyclotomic rings (fast, easy to implement).
   - ⋆ Cf. [BGV'12]: $\tilde{O}(\lambda)$ amortized across λ ciphertexts, exotic rings.

Our Results

Practical bootstrapping algorithms with quasi-linear $\tilde{O}(\lambda)$ runtimes:

1. For "unpacked" (single-bit) plaintexts:
   - ✔ Extremely simple!
   - ✔ Uses only power-of-2 cyclotomic rings (fast, easy to implement).
   - ⋆ Cf. [BGV'12]: $\tilde{O}(\lambda)$ amortized across λ ciphertexts, exotic rings.

2. For "packed" (many-bit) plaintexts:

Our Results

Practical bootstrapping algorithms with quasi-linear $\tilde{O}(\lambda)$ runtimes:

1. For "unpacked" (single-bit) plaintexts:
   - ✔ Extremely simple!
   - ✔ Uses only power-of-2 cyclotomic rings (fast, easy to implement).
   - ⋆ Cf. [BGV'12]: $\tilde{O}(\lambda)$ amortized across λ ciphertexts, exotic rings.
2. For "packed" (many-bit) plaintexts:
   - ⋆ Based on a substantial enhancement of "ring-switching" [GHPS'12] to non-subrings.

Our Results

Practical bootstrapping algorithms with quasi-linear $\tilde{O}(\lambda)$ runtimes:

1. For "unpacked" (single-bit) plaintexts:
   - ✔ Extremely simple!
   - ✔ Uses only power-of-2 cyclotomic rings (fast, easy to implement).
   - ⋆ Cf. [BGV'12]: $\tilde{O}(\lambda)$ amortized across λ ciphertexts, exotic rings.

2. For "packed" (many-bit) plaintexts:
   - ⋆ Based on a substantial enhancement of "ring-switching" [GHPS'12] to non-subrings.
   - ✔ Appears quite practical, avoids both main inefficiencies of [GHS'12b]: no homomorphic reduction modulo $\Phi_m(X)$, no generic compilation.

Our Results

Practical bootstrapping algorithms with quasi-linear $\tilde{O}(\lambda)$ runtimes:

1. For "unpacked" (single-bit) plaintexts:
   - ✔ Extremely simple!
   - ✔ Uses only power-of-2 cyclotomic rings (fast, easy to implement).
   - ⋆ Cf. [BGV'12]: $\tilde{O}(\lambda)$ amortized across λ ciphertexts, exotic rings.

2. For "packed" (many-bit) plaintexts:
   - ⋆ Based on a substantial enhancement of "ring-switching" [GHPS'12] to non-subrings.
   - ✔ Appears quite practical, avoids both main inefficiencies of [GHS'12b]: no homomorphic reduction modulo $\Phi_m(X)$, no generic compilation.
   - ✔ Special purpose, completely algebraic description – no "circuits."

Our Results

Practical bootstrapping algorithms with quasi-linear $\tilde{O}(\lambda)$ runtimes:

1. For "unpacked" (single-bit) plaintexts:
   - ✔ Extremely simple!
   - ✔ Uses only power-of-2 cyclotomic rings (fast, easy to implement).
   - ⋆ Cf. [BGV'12]: $\tilde{O}(\lambda)$ amortized across λ ciphertexts, exotic rings.

2. For "packed" (many-bit) plaintexts:
   - ⋆ Based on a substantial enhancement of "ring-switching" [GHPS'12] to non-subrings.
   - ✔ Appears quite practical, avoids both main inefficiencies of [GHS'12b]: no homomorphic reduction modulo $\Phi_m(X)$, no generic compilation.
   - ✔ Special purpose, completely algebraic description – no "circuits."
   - ✔ Completely decouples the algebraic structure of SHE plaintext ring from that needed for bootstrapping.

Setting the Stage: Decryption in SHE [LPR'10,BV'11,BGV'12]

- Let $R = \mathbb{Z}[X]/(X^{k/2}+1)$, for k a power of 2. (The kth cyclotomic ring.)

Setting the Stage: Decryption in SHE [LPR'10,BV'11,BGV'12]

- Let $R = \mathbb{Z}[X]/(X^{k/2}+1)$, for k a power of 2. (The kth cyclotomic ring.)

  Let $R_q = R/qR = \mathbb{Z}_q[X]/(X^{k/2}+1)$ for any integer q.

Setting the Stage: Decryption in SHE [LPR'10,BV'11,BGV'12]

- Let $R = \mathbb{Z}[X]/(X^{k/2} + 1)$, for k a power of 2. (The kth cyclotomic ring.)
  Let $R_q = R/qR = \mathbb{Z}_q[X]/(X^{k/2} + 1)$ for any integer q.
- Plaintext ring is R_2, ciphertext ring is R_q for $q \gg 2$.
  Can assume $k, q = \tilde{O}(\lambda)$ by ring- and modulus-switching.

Setting the Stage: Decryption in SHE [LPR'10,BV'11,BGV'12]

- Let $R = \mathbb{Z}[X]/(X^{k/2}+1)$, for k a power of 2. (The kth cyclotomic ring.)
  Let $R_q = R/qR = \mathbb{Z}_q[X]/(X^{k/2}+1)$ for any integer q.
- Plaintext ring is R_2, ciphertext ring is R_q for $q \gg 2$.
  Can assume $k, q = \tilde{O}(\lambda)$ by ring- and modulus-switching.
- Ciphertext $c = (c_0, c_1) \in R_q^2$ encrypting $\mu \in R_2$ under $s \in R$ satisfies

$$v = c_0 + c_1 \cdot s \approx \frac{q}{2}\mu \pmod{qR}.$$

Setting the Stage: Decryption in SHE [LPR'10,BV'11,BGV'12]

- Let $R = \mathbb{Z}[X]/(X^{k/2}+1)$, for k a power of 2. (The kth cyclotomic ring.)
  Let $R_q = R/qR = \mathbb{Z}_q[X]/(X^{k/2}+1)$ for any integer q.
- Plaintext ring is R_2, ciphertext ring is R_q for $q \gg 2$.
  Can assume $k, q = \tilde{O}(\lambda)$ by ring- and modulus-switching.
- Ciphertext $c = (c_0, c_1) \in R_q^2$ encrypting $\mu \in R_2$ under $s \in R$ satisfies

  $$v = c_0 + c_1 \cdot s \approx \tfrac{q}{2}\mu \pmod{qR}.$$

  Define the decryption function

  $$\mathsf{Dec}_s(c) := \lfloor v \rceil = \mu \in R_2,$$

  where "rounding" $\lfloor \cdot \rceil \colon \mathbb{Z}_q \to \mathbb{Z}_2$ is applied to coeffs of $v = v(X)$.

Setting the Stage: Decryption in SHE [LPR'10,BV'11,BGV'12]

- Let $R = \mathbb{Z}[X]/(X^{k/2}+1)$, for k a power of 2. (The kth cyclotomic ring.)
  Let $R_q = R/qR = \mathbb{Z}_q[X]/(X^{k/2}+1)$ for any integer q.
- Plaintext ring is R_2, ciphertext ring is R_q for $q \gg 2$.
  Can assume $k, q = \tilde{O}(\lambda)$ by ring- and modulus-switching.
- Ciphertext $c = (c_0, c_1) \in R_q^2$ encrypting $\mu \in R_2$ under $s \in R$ satisfies

  $$v = c_0 + c_1 \cdot s \approx \tfrac{q}{2}\mu \pmod{qR}.$$

  Define the decryption function

  $$\mathsf{Dec}_s(c) := \lfloor v \rceil = \mu \in R_2,$$

  where "rounding" $\lfloor\cdot\rceil \colon \mathbb{Z}_q \to \mathbb{Z}_2$ is applied to coeffs of $v = v(X)$.
- "Unpacked" plaintext $\mu \in \mathbb{Z}_2 \subseteq R_2$, i.e., just a constant polynomial.

Setting the Stage: Decryption in SHE [LPR'10,BV'11,BGV'12]

- Let $R = \mathbb{Z}[X]/(X^{k/2}+1)$, for k a power of 2. (The kth cyclotomic ring.)
  Let $R_q = R/qR = \mathbb{Z}_q[X]/(X^{k/2}+1)$ for any integer q.
- Plaintext ring is R_2, ciphertext ring is R_q for $q \gg 2$.
  Can assume $k, q = \tilde{O}(\lambda)$ by ring- and modulus-switching.
- Ciphertext $c = (c_0, c_1) \in R_q^2$ encrypting $\mu \in R_2$ under $s \in R$ satisfies

  $$v = c_0 + c_1 \cdot s \approx \tfrac{q}{2}\mu \pmod{qR}.$$

  Define the decryption function

  $$\mathsf{Dec}_s(c) := \lfloor v \rceil = \mu \in R_2,$$

  where "rounding" $\lfloor \cdot \rceil \colon \mathbb{Z}_q \to \mathbb{Z}_2$ is applied to coeffs of $v = v(X)$.
- "Unpacked" plaintext $\mu \in \mathbb{Z}_2 \subseteq R_2$, i.e., just a constant polynomial.
  "Packed" plaintext uses more of R_2, e.g., multiple "slots" [SV'11].

Warm-Up:
Bootstrapping Unpacked Ciphertexts

Bootstrapping Unpacked Ciphertexts: Main Idea

1. Isolate message-carrying coefficient v_0 of $v(X)$ by homomorphically "tracing down" a tower of cyclotomic rings $\mathcal{O}_{2k}/\mathcal{O}_k/\cdots/\mathcal{O}_4/\mathbb{Z}$.
   (Trace $=$ sum of the two automorphisms of $\mathcal{O}_{2i}/\mathcal{O}_i$.)

Bootstrapping Unpacked Ciphertexts: Main Idea

1. Isolate message-carrying coefficient v_0 of $v(X)$ by homomorphically "tracing down" a tower of cyclotomic rings $\mathcal{O}_{2k}/\mathcal{O}_k/\cdots/\mathcal{O}_4/\mathbb{Z}$. (Trace $=$ sum of the two automorphisms of $\mathcal{O}_{2i}/\mathcal{O}_i$.)

$$v_0 + v_1 X + v_2 X^2 + \cdots v_{k-1} X^{k-1} \qquad \mathbb{Z}_q[X]/(X^k+1)$$

$$v_0 + 0X + v_2 X^2 + \cdots 0 X^{k-1} \qquad \mathbb{Z}_q[X^2]/(X^k+1)$$

$$\vdots$$

$$v_0 + v_{k/4} X^{k/4} + \cdots + v_{3k/4} X^{3k/4} \qquad \mathbb{Z}_q[X^{k/4}]/(X^k+1)$$

$$v_0 + v_{k/2} X^{k/2} \qquad \mathbb{Z}_q[X^{k/2}]/(X^k+1)$$

$$v_0 \qquad \mathbb{Z}_q$$

Bootstrapping Unpacked Ciphertexts: Main Idea

1. Isolate message-carrying coefficient v_0 of $v(X)$ by homomorphically "tracing down" a tower of cyclotomic rings $\mathcal{O}_{2k}/\mathcal{O}_k/\cdots/\mathcal{O}_4/\mathbb{Z}$. (Trace $=$ sum of the two automorphisms of $\mathcal{O}_{2i}/\mathcal{O}_i$.)

$$\begin{array}{cc}
v_0 + v_1 X + v_2 X^2 + \cdots v_{k-1} X^{k-1} & \mathbb{Z}_q[X]/(X^k+1) \\
 & | \\
v_0 + 0X + v_2 X^2 + \cdots 0X^{k-1} & \mathbb{Z}_q[X^2]/(X^k+1) \\
 & \vdots \\
v_0 + v_{k/4} X^{k/4} + \cdots + v_{3k/4} X^{3k/4} & \mathbb{Z}_q[X^{k/4}]/(X^k+1) \\
 & | \\
v_0 + v_{k/2} X^{k/2} & \mathbb{Z}_q[X^{k/2}]/(X^k+1) \\
 & | \\
v_0 & \mathbb{Z}_q
\end{array}$$

2. Homomorphically "round" $v_0 \in \mathbb{Z}_q$ to the message bit $\lfloor \frac{2}{q} \cdot v_0 \rceil \in \mathbb{Z}_2$.

Algebra: Cyclotomic Towers and Product Bases

- Let $\zeta = \zeta_k$ have order k, a power of 2. Its min. poly: $\zeta^{k/2} + 1 = 0$.

Algebra: Cyclotomic Towers and Product Bases

- Let $\zeta = \zeta_k$ have order k, a power of 2. Its min. poly: $\zeta^{k/2} + 1 = 0$.
  So $\mathcal{O}_k = \mathbb{Z}[\zeta] \cong \mathbb{Z}[X]/(X^{k/2} + 1)$ has $\mathbb{Z}$-basis $\{1, \zeta, \zeta^2, \ldots, \zeta^{k/2-1}\}$.

Algebra: Cyclotomic Towers and Product Bases

- Let $\zeta = \zeta_k$ have order k, a power of 2. Its min. poly: $\zeta^{k/2} + 1 = 0$. So $\mathcal{O}_k = \mathbb{Z}[\zeta] \cong \mathbb{Z}[X]/(X^{k/2} + 1)$ has $\mathbb{Z}$-basis $\{1, \zeta, \zeta^2, \ldots, \zeta^{k/2-1}\}$.
- Tower of quadratic extensions $\mathcal{O}_k / \mathcal{O}_{k/2} / \cdots / \mathcal{O}_4 / \mathbb{Z}$:

Algebra: Cyclotomic Towers and Product Bases

- Let $\zeta = \zeta_k$ have order k, a power of 2. Its min. poly: $\zeta^{k/2} + 1 = 0$. So $\mathcal{O}_k = \mathbb{Z}[\zeta] \cong \mathbb{Z}[X]/(X^{k/2}+1)$ has $\mathbb{Z}$-basis $\{1, \zeta, \zeta^2, \ldots, \zeta^{k/2-1}\}$.
- Tower of quadratic extensions $\mathcal{O}_k/\mathcal{O}_{k/2}/\cdots/\mathcal{O}_4/\mathbb{Z}$:

$$
\begin{array}{ccc}
\zeta_k^2 = \zeta_{k/2} & \mathcal{O}_k = \mathcal{O}_{k/2}[\zeta_k] & \mathcal{O}_{k/2}\text{-basis } B'_k = \{1, \zeta_k\} \\
 & \vdots & \\
\zeta_8^2 = \zeta_4 & \mathcal{O}_8 = \mathcal{O}_4[\zeta_8] & \mathcal{O}_4\text{-basis } B'_8 = \{1, \zeta_8\} \\
 & | & \\
\zeta_4^2 = \zeta_2 & \mathcal{O}_4 = \mathcal{O}_2[\zeta_4] & \mathcal{O}_2\text{-basis } B'_4 = \{1, \zeta_4\} \\
 & | & \\
\zeta_2^2 = 1 & \mathcal{O}_2 = \mathbb{Z}[\zeta_2] = \mathbb{Z} & \mathbb{Z}\text{-basis } B'_2 = \{1\}
\end{array}
$$

Algebra: Cyclotomic Towers and Product Bases

- Let $\zeta = \zeta_k$ have order k, a power of 2. Its min. poly: $\zeta^{k/2} + 1 = 0$. So $\mathcal{O}_k = \mathbb{Z}[\zeta] \cong \mathbb{Z}[X]/(X^{k/2}+1)$ has $\mathbb{Z}$-basis $\{1, \zeta, \zeta^2, \ldots, \zeta^{k/2-1}\}$.
- Tower of quadratic extensions $\mathcal{O}_k/\mathcal{O}_{k/2}/\cdots/\mathcal{O}_4/\mathbb{Z}$:

$$\begin{array}{ccc}
\zeta_k^2 = \zeta_{k/2} & \mathcal{O}_k = \mathcal{O}_{k/2}[\zeta_k] & \mathcal{O}_{k/2}\text{-basis } B'_k = \{1, \zeta_k\} \\
 & \vdots & \\
\zeta_8^2 = \zeta_4 & \mathcal{O}_8 = \mathcal{O}_4[\zeta_8] & \mathcal{O}_4\text{-basis } B'_8 = \{1, \zeta_8\} \\
 & | & \\
\zeta_4^2 = \zeta_2 & \mathcal{O}_4 = \mathcal{O}_2[\zeta_4] & \mathcal{O}_2\text{-basis } B'_4 = \{1, \zeta_4\} \\
 & | & \\
\zeta_2^2 = 1 & \mathcal{O}_2 = \mathbb{Z}[\zeta_2] = \mathbb{Z} & \mathbb{Z}\text{-basis } B'_2 = \{1\}
\end{array}$$

- "Product" $\mathbb{Z}$-basis of $\mathcal{O}_k$:

$$B_k := B'_k \cdot B_{k/2} = B'_k \cdot B'_{k/2} \cdots B'_2 = \{1, \zeta, \zeta^2, \ldots, \zeta^{k/2-1}\}.$$

Algebra: The Trace

- Tower of quadratic extensions $\mathcal{O}_k/\mathcal{O}_{k/2}/\cdots/\mathcal{O}_4/\mathbb{Z}$, where $\zeta_i^2 = \zeta_{i/2}$.

Algebra: The Trace

- Tower of quadratic extensions $\mathcal{O}_k/\mathcal{O}_{k/2}/\cdots/\mathcal{O}_4/\mathbb{Z}$, where $\zeta_i^2 = \zeta_{i/2}$.
- $\mathcal{O}_i$ has exactly two automorphisms that fix $\mathcal{O}_{i/2}$: $\zeta_i \mapsto \pm\,\zeta_i$.

  The trace function $\mathrm{Tr}\colon \mathcal{O}_i \to \mathcal{O}_{i/2}$ simply sums these automorphisms.

Algebra: The Trace

- Tower of quadratic extensions $\mathcal{O}_k/\mathcal{O}_{k/2}/\cdots/\mathcal{O}_4/\mathbb{Z}$, where $\zeta_i^2 = \zeta_{i/2}$.
- $\mathcal{O}_i$ has exactly two automorphisms that fix $\mathcal{O}_{i/2}$: $\zeta_i \mapsto \pm\,\zeta_i$.

  The trace function $\operatorname{Tr}\colon \mathcal{O}_i \to \mathcal{O}_{i/2}$ simply sums these automorphisms.
- Let $v = v_0 \cdot 1 + v_1 \cdot \zeta_i \in \mathcal{O}_i$ for $v_0, v_1 \in \mathcal{O}_{i/2}$.

  Then $\operatorname{Tr}(v) = 2 \cdot v_0$. So $\operatorname{Tr}(\mathcal{O}_i) = 2 \cdot \mathcal{O}_{i/2}$.

Algebra: The Trace

- Tower of quadratic extensions $\mathcal{O}_k/\mathcal{O}_{k/2}/\cdots/\mathcal{O}_4/\mathbb{Z}$, where $\zeta_i^2 = \zeta_{i/2}$.
- $\mathcal{O}_i$ has exactly two automorphisms that fix $\mathcal{O}_{i/2}$: $\zeta_i \mapsto \pm\,\zeta_i$.

  The trace function $\mathrm{Tr}\colon \mathcal{O}_i \to \mathcal{O}_{i/2}$ simply sums these automorphisms.
- Let $v = v_0 \cdot 1 + v_1 \cdot \zeta_i \in \mathcal{O}_i$ for $v_0, v_1 \in \mathcal{O}_{i/2}$.

  Then $\mathrm{Tr}(v) = 2 \cdot v_0$. So $\mathrm{Tr}(\mathcal{O}_i) = 2 \cdot \mathcal{O}_{i/2}$.
- More generally, $\mathrm{Tr}_{\mathcal{O}_i/\mathcal{O}_{i'}}$ sums the automorphisms of $\mathcal{O}_i$ that fix $\mathcal{O}_{i'}$.
  Key facts:

Algebra: The Trace

- Tower of quadratic extensions $\mathcal{O}_k/\mathcal{O}_{k/2}/\cdots/\mathcal{O}_4/\mathbb{Z}$, where $\zeta_i^2 = \zeta_{i/2}$.
- $\mathcal{O}_i$ has exactly two automorphisms that fix $\mathcal{O}_{i/2}$: $\zeta_i \mapsto \pm\zeta_i$.

  The trace function $\mathrm{Tr}\colon \mathcal{O}_i \to \mathcal{O}_{i/2}$ simply sums these automorphisms.
- Let $v = v_0 \cdot 1 + v_1 \cdot \zeta_i \in \mathcal{O}_i$ for $v_0, v_1 \in \mathcal{O}_{i/2}$.

  Then $\mathrm{Tr}(v) = 2 \cdot v_0$. So $\mathrm{Tr}(\mathcal{O}_i) = 2 \cdot \mathcal{O}_{i/2}$.
- More generally, $\mathrm{Tr}_{\mathcal{O}_i/\mathcal{O}_{i'}}$ sums the automorphisms of $\mathcal{O}_i$ that fix $\mathcal{O}_{i'}$. Key facts:
  - ⋆ $\mathrm{Tr}_{\mathcal{O}_i/\mathcal{O}_{i''}} = \mathrm{Tr}_{\mathcal{O}_{i'}/\mathcal{O}_{i''}} \circ \mathrm{Tr}_{\mathcal{O}_i/\mathcal{O}_{i'}}$

Algebra: The Trace

- Tower of quadratic extensions $\mathcal{O}_k/\mathcal{O}_{k/2}/\cdots/\mathcal{O}_4/\mathbb{Z}$, where $\zeta_i^2 = \zeta_{i/2}$.
- $\mathcal{O}_i$ has exactly two automorphisms that fix $\mathcal{O}_{i/2}$: $\zeta_i \mapsto \pm\zeta_i$.

  The trace function $\mathrm{Tr}\colon \mathcal{O}_i \to \mathcal{O}_{i/2}$ simply sums these automorphisms.
- Let $v = v_0 \cdot 1 + v_1 \cdot \zeta_i \in \mathcal{O}_i$ for $v_0, v_1 \in \mathcal{O}_{i/2}$.

  Then $\mathrm{Tr}(v) = 2 \cdot v_0$. So $\mathrm{Tr}(\mathcal{O}_i) = 2 \cdot \mathcal{O}_{i/2}$.
- More generally, $\mathrm{Tr}_{\mathcal{O}_i/\mathcal{O}_{i'}}$ sums the automorphisms of $\mathcal{O}_i$ that fix $\mathcal{O}_{i'}$. Key facts:

  ⋆ $\mathrm{Tr}_{\mathcal{O}_i/\mathcal{O}_{i''}} = \mathrm{Tr}_{\mathcal{O}_{i'}/\mathcal{O}_{i''}} \circ \mathrm{Tr}_{\mathcal{O}_i/\mathcal{O}_{i'}}$

  $\Rightarrow$ $\mathrm{Tr}_{\mathcal{O}_i/\mathcal{O}_{i'}}(\mathcal{O}_i) = \deg(\mathcal{O}_i/\mathcal{O}_{i'}) \cdot \mathcal{O}_{i'}$.

Algebra: The Trace

- Tower of quadratic extensions $\mathcal{O}_k/\mathcal{O}_{k/2}/\cdots/\mathcal{O}_4/\mathbb{Z}$, where $\zeta_i^2 = \zeta_{i/2}$.
- $\mathcal{O}_i$ has exactly two automorphisms that fix $\mathcal{O}_{i/2}$: $\zeta_i \mapsto \pm\zeta_i$.

  The trace function $\mathrm{Tr}\colon \mathcal{O}_i \to \mathcal{O}_{i/2}$ simply sums these automorphisms.
- Let $v = v_0 \cdot 1 + v_1 \cdot \zeta_i \in \mathcal{O}_i$ for $v_0, v_1 \in \mathcal{O}_{i/2}$.

  Then $\mathrm{Tr}(v) = 2 \cdot v_0$. So $\mathrm{Tr}(\mathcal{O}_i) = 2 \cdot \mathcal{O}_{i/2}$.
- More generally, $\mathrm{Tr}_{\mathcal{O}_i/\mathcal{O}_{i'}}$ sums the automorphisms of $\mathcal{O}_i$ that fix $\mathcal{O}_{i'}$. Key facts:

  ⋆ $\mathrm{Tr}_{\mathcal{O}_i/\mathcal{O}_{i''}} = \mathrm{Tr}_{\mathcal{O}_{i'}/\mathcal{O}_{i''}} \circ \mathrm{Tr}_{\mathcal{O}_i/\mathcal{O}_{i'}}$

  $\Rightarrow$ $\mathrm{Tr}_{\mathcal{O}_i/\mathcal{O}_{i'}}(\mathcal{O}_i) = \deg(\mathcal{O}_i/\mathcal{O}_{i'}) \cdot \mathcal{O}_{i'}$.

  $\Rightarrow$ $\mathrm{Tr}_{\mathcal{O}_i/\mathbb{Z}}(v) = \frac{i}{2} \cdot v_0$, where $v_0 \in \mathbb{Z}$ is the coeff of $\zeta_i^0 = 1$.

Bootstrapping Unpacked Ciphertexts: Overview

Recall: $R = \mathcal{O}_k$, and $v = c_0 + c_1 \cdot s \approx \frac{q}{2}\mu \in R_q$ for message $\mu \in \mathbb{Z}_2 \subseteq R_2$.

Bootstrapping Unpacked Ciphertexts: Overview

Recall: $R = \mathcal{O}_k$, and $v = c_0 + c_1 \cdot s \approx \frac{q}{2}\mu \in R_q$ for message $\mu \in \mathbb{Z}_2 \subseteq R_2$.

1. Prepare:
   - View c as a "noiseless" encryption of plaintext

     $$v = \frac{q}{q} \cdot v + 0 = c_0 + c_1 \cdot s \in R_q.$$

     Plaintext ring is now R_q, not R_2!

Bootstrapping Unpacked Ciphertexts: Overview

Recall: $R = \mathcal{O}_k$, and $v = c_0 + c_1 \cdot s \approx \frac{q}{2}\mu \in R_q$ for message $\mu \in \mathbb{Z}_2 \subseteq R_2$.

1. Prepare:
   - ⋆ View c as a "noiseless" encryption of plaintext

     $$v = \tfrac{q}{q} \cdot v + 0 = c_0 + c_1 \cdot s \in R_q.$$

     Plaintext ring is now R_q, not R_2!
   - ⋆ (Switch to larger ciphertext modulus $Q \gg q$ and ring $\tilde{R} \supseteq R$, to support upcoming homomorphic operations.)

Bootstrapping Unpacked Ciphertexts: Overview

Recall: $R = \mathcal{O}_k$, and $v = c_0 + c_1 \cdot s \approx \frac{q}{2}\mu \in R_q$ for message $\mu \in \mathbb{Z}_2 \subseteq R_2$.

1. Prepare:
   - View c as a "noiseless" encryption of plaintext
     $$v = \tfrac{q}{q} \cdot v + 0 = c_0 + c_1 \cdot s \in R_q.$$
     Plaintext ring is now R_q, not R_2!
   - (Switch to larger ciphertext modulus $Q \gg q$ and ring $\tilde{R} \supseteq R$, to support upcoming homomorphic operations.)
2. Extract "constant term" $v_0 \in \mathbb{Z}_q$ of v: homomorphically evaluate
   $$\frac{\mathrm{Tr}_{R/\mathbb{Z}}(v)}{\deg(R/\mathbb{Z})} = v_0 \approx \tfrac{q}{2} \cdot \mu \in \mathbb{Z}_q.$$
   Fast, increases noise rate by only $\approx \sqrt{k}$ factor.

Bootstrapping Unpacked Ciphertexts: Overview

Recall: $R = \mathcal{O}_k$, and $v = c_0 + c_1 \cdot s \approx \frac{q}{2}\mu \in R_q$ for message $\mu \in \mathbb{Z}_2 \subseteq R_2$.

1. Prepare:
   - ★ View c as a "noiseless" encryption of plaintext
     $$v = \tfrac{q}{q} \cdot v + 0 = c_0 + c_1 \cdot s \in R_q.$$
     Plaintext ring is now R_q, not R_2!
   - ★ (Switch to larger ciphertext modulus $Q \gg q$ and ring $\tilde{R} \supseteq R$, to support upcoming homomorphic operations.)
2. Extract "constant term" $v_0 \in \mathbb{Z}_q$ of v: homomorphically evaluate
   $$\frac{\mathrm{Tr}_{R/\mathbb{Z}}(v)}{\deg(R/\mathbb{Z})} = v_0 \approx \tfrac{q}{2} \cdot \mu \in \mathbb{Z}_q.$$
   Fast, increases noise rate by only $\approx \sqrt{k}$ factor.
3. Round: homomorphically evaluate $\lfloor v_0 \rceil = \mu \in \mathbb{Z}_2$.

   Uses algebraic procedure of depth $\lg(q/2)$ & size $\lg^2(q/2)$ [GHS'12b]

Bootstrapping Unpacked Ciphertexts: Overview

Recall: $R = \mathcal{O}_k$, and $v = c_0 + c_1 \cdot s \approx \frac{q}{2}\mu \in R_q$ for message $\mu \in \mathbb{Z}_2 \subseteq R_2$.

1. Prepare:
   - ⋆ View c as a "noiseless" encryption of plaintext
   $$v = \tfrac{q}{q} \cdot v + 0 = c_0 + c_1 \cdot s \in R_q.$$
   Plaintext ring is now R_q, not R_2!
   - ⋆ (Switch to larger ciphertext modulus $Q \gg q$ and ring $\tilde{R} \supseteq R$, to support upcoming homomorphic operations.)
2. Extract "constant term" $v_0 \in \mathbb{Z}_q$ of v: homomorphically evaluate
   $$\frac{\mathrm{Tr}_{R/\mathbb{Z}}(v)}{\deg(R/\mathbb{Z})} = v_0 \approx \tfrac{q}{2} \cdot \mu \in \mathbb{Z}_q.$$
   Fast, increases noise rate by only $\approx \sqrt{k}$ factor.
3. Round: homomorphically evaluate $\lfloor v_0 \rceil = \mu \in \mathbb{Z}_2$.
   Uses algebraic procedure of depth $\lg(q/2)$ & size $\lg^2(q/2)$ [GHS'12b]

⋆⋆ Now have an encryption of $\lfloor v_0 \rceil = \mu$. Done!

Evaluating Trace$_{R/\mathbb{Z}}$ Homomorphically

?? Use "ring switching" [GHPS'12] ?

- ✔ Computes $\mathrm{Tr}_{R/R'}$ homomorphically, by taking $\mathrm{Tr}_{R/R'}$ of ciphertext.

Evaluating $\text{Trace}_{R/\mathbb{Z}}$ Homomorphically

?? Use "ring switching" [GHPS'12] ?

- ✔ Computes $\mathrm{Tr}_{R/R'}$ homomorphically, by taking $\mathrm{Tr}_{R/R'}$ of ciphertext.
- ✗ Requires hardness of ring-LWE in R' ... but here $R' = \mathbb{Z}$.

Evaluating Trace$_{R/\mathbb{Z}}$ Homomorphically

?? Use "ring switching" [GHPS'12] ?

- ✔ Computes $\mathrm{Tr}_{R/R'}$ homomorphically, by taking $\mathrm{Tr}_{R/R'}$ of ciphertext.
- ✗ Requires hardness of ring-LWE in R' ... but here $R' = \mathbb{Z}$.

?? Directly apply all automorphisms τ of $R/\mathbb{Z}$ to ciphertext, then sum?

$$\tau(c_0) + \tau(c_1) \cdot \tau(s) = \tau(v) \quad \overset{\text{key-switch}}{\Longrightarrow} \quad c_0' + c_1' \cdot s \approx \tau(v)$$

Evaluating $\mathrm{Trace}_{R/\mathbb{Z}}$ Homomorphically

?? Use "ring switching" [GHPS'12] ?

- ✔ Computes $\mathrm{Tr}_{R/R'}$ homomorphically, by taking $\mathrm{Tr}_{R/R'}$ of ciphertext.
- ✗ Requires hardness of ring-LWE in R' ... but here $R' = \mathbb{Z}$.

?? Directly apply all automorphisms τ of $R/\mathbb{Z}$ to ciphertext, then sum?

$$\tau(c_0) + \tau(c_1) \cdot \tau(s) = \tau(v) \quad \overset{\text{key-switch}}{\Longrightarrow} \quad c_0' + c_1' \cdot s \approx \tau(v)$$

- ✗ $k/2$ automorphisms & key-switches: quadratic work & space

Evaluating Trace$_{R/\mathbb{Z}}$ Homomorphically

?? Use "ring switching" [GHPS'12] ?

- ✔ Computes $\mathrm{Tr}_{R/R'}$ homomorphically, by taking $\mathrm{Tr}_{R/R'}$ of ciphertext.
- ✗ Requires hardness of ring-LWE in R' ... but here $R' = \mathbb{Z}$.

?? Directly apply all automorphisms τ of $R/\mathbb{Z}$ to ciphertext, then sum?

$$\tau(c_0) + \tau(c_1) \cdot \tau(s) = \tau(v) \quad \overset{\text{key-switch}}{\Longrightarrow} \quad c_0' + c_1' \cdot s \approx \tau(v)$$

- ✗ $k/2$ automorphisms & key-switches: quadratic work & space

✔ Iteratively "trace down" $R = \mathcal{O}_k \to \mathcal{O}_{k/2} \to \cdots \to \mathbb{Z}$.

Evaluating $\text{Trace}_{R/\mathbb{Z}}$ Homomorphically

?? Use "ring switching" [GHPS'12] ?

- ✔ Computes $\mathrm{Tr}_{R/R'}$ homomorphically, by taking $\mathrm{Tr}_{R/R'}$ of ciphertext.
- ✗ Requires hardness of ring-LWE in R' ... but here $R' = \mathbb{Z}$.

?? Directly apply all automorphisms τ of $R/\mathbb{Z}$ to ciphertext, then sum?

$$\tau(c_0) + \tau(c_1) \cdot \tau(s) = \tau(v) \quad \overset{\text{key-switch}}{\Longrightarrow} \quad c_0' + c_1' \cdot s \approx \tau(v)$$

- ✗ $k/2$ automorphisms & key-switches: quadratic work & space

✔ Iteratively "trace down" $R = \mathcal{O}_k \to \mathcal{O}_{k/2} \to \cdots \to \mathbb{Z}$.

- ⋆ Only need to apply the two automorphisms of each $\mathcal{O}_i/\mathcal{O}_{i/2}$.
- ⋆ Total $\lg(k)$ automorphisms & key-switches $\Rightarrow \tilde{O}(k)$ work.

Evaluating Trace$_{R/\mathbb{Z}}$ Homomorphically

?? Use "ring switching" [GHPS'12] ?

- ✔ Computes $\mathrm{Tr}_{R/R'}$ homomorphically, by taking $\mathrm{Tr}_{R/R'}$ of ciphertext.
- ✗ Requires hardness of ring-LWE in R' ... but here $R' = \mathbb{Z}$.

?? Directly apply all automorphisms τ of $R/\mathbb{Z}$ to ciphertext, then sum?

$$\tau(c_0) + \tau(c_1) \cdot \tau(s) = \tau(v) \quad \stackrel{\text{key-switch}}{\Longrightarrow} \quad c_0' + c_1' \cdot s \approx \tau(v)$$

- ✗ $k/2$ automorphisms & key-switches: quadratic work & space

✔ Iteratively "trace down" $R = \mathcal{O}_k \to \mathcal{O}_{k/2} \to \cdots \to \mathbb{Z}$.

- ⋆ Only need to apply the two automorphisms of each $\mathcal{O}_i/\mathcal{O}_{i/2}$.
- ⋆ Total $\lg(k)$ automorphisms & key-switches $\Rightarrow \tilde{O}(k)$ work.

Detail #1: ciphertexts are over $\tilde{R} \supseteq R$, so use automorphisms of $\tilde{R}$ that coincide with those of $\mathcal{O}_i/\mathcal{O}_{i/2}$.

Evaluating $\mathrm{Trace}_{R/\mathbb{Z}}$ Homomorphically

?? Use "ring switching" [GHPS'12] ?

- ✔ Computes $\mathrm{Tr}_{R/R'}$ homomorphically, by taking $\mathrm{Tr}_{R/R'}$ of ciphertext.
- ✗ Requires hardness of ring-LWE in R' . . . but here $R' = \mathbb{Z}$.

?? Directly apply all automorphisms τ of $R/\mathbb{Z}$ to ciphertext, then sum?

$$\tau(c_0) + \tau(c_1) \cdot \tau(s) = \tau(v) \quad \overset{\text{key-switch}}{\Longrightarrow} \quad c_0' + c_1' \cdot s \approx \tau(v)$$

- ✗ $k/2$ automorphisms & key-switches: quadratic work & space

✔ Iteratively "trace down" $R = \mathcal{O}_k \to \mathcal{O}_{k/2} \to \cdots \to \mathbb{Z}$.

- ⋆ Only need to apply the two automorphisms of each $\mathcal{O}_i/\mathcal{O}_{i/2}$.
- ⋆ Total $\lg(k)$ automorphisms & key-switches $\Rightarrow \tilde{O}(k)$ work.

Detail #1: ciphertexts are over $\tilde{R} \supseteq R$, so use automorphisms of $\tilde{R}$ that coincide with those of $\mathcal{O}_i/\mathcal{O}_{i/2}$.

Detail #2: each $\mathrm{Tr}(\mathcal{O}_i) = 2\mathcal{O}_{i/2}$, so lift to plaintext modulus $2q$, then halve result.

Main Result:
Bootstrapping Packed Ciphertexts

Bootstrapping Packed Ciphertexts: Overview

1. Prepare: as before, view c as a "noiseless" encryption of plaintext

$$v = c_0 + c_1 \cdot s = \sum_j v_j \cdot b_j \in R_q.$$

Recall: $\mu = \lfloor v \rceil = \sum_j \lfloor v_j \rceil \cdot b_j \in R_2$ (where $b_j = \zeta^j$).

Bootstrapping Packed Ciphertexts: Overview

1. Prepare: as before, view c as a "noiseless" encryption of plaintext

$$v = c_0 + c_1 \cdot s = \sum_j v_j \cdot b_j \in R_q.$$

Recall: $\mu = \lfloor v \rceil = \sum_j \lfloor v_j \rceil \cdot b_j \in R_2$ (where $b_j = \zeta^j$).

2. Homomorphically map coeffs v_j to "$\mathbb{Z}_q$-slots" of certain ring S_q:

$$\sum v_j \cdot b_j \in R_q \quad \longmapsto \quad \sum v_j \cdot c_j \in S_q.$$

(Change of basis, analogous to homomorphic DFT.)

Bootstrapping Packed Ciphertexts: Overview

1. Prepare: as before, view c as a "noiseless" encryption of plaintext

$$v = c_0 + c_1 \cdot s = \sum_j v_j \cdot b_j \in R_q.$$

   Recall: $\mu = \lfloor v \rceil = \sum_j \lfloor v_j \rceil \cdot b_j \in R_2$ (where $b_j = \zeta^j$).

2. Homomorphically map coeffs v_j to "$\mathbb{Z}_q$-slots" of certain ring S_q:

$$\sum v_j \cdot b_j \in R_q \quad \longmapsto \quad \sum v_j \cdot c_j \in S_q.$$

   (Change of basis, analogous to homomorphic DFT.)

3. Batch-round: homom'ly apply $\lfloor \cdot \rceil$ on all $\mathbb{Z}_q$-slots at once [SV'11]:

$$\sum v_j \cdot c_j \in S_q \quad \longmapsto \quad \sum \lfloor v_j \rceil \cdot c_j \in S_2.$$

Bootstrapping Packed Ciphertexts: Overview

1. Prepare: as before, view c as a "noiseless" encryption of plaintext

$$v = c_0 + c_1 \cdot s = \sum_j v_j \cdot b_j \in R_q.$$

Recall: $\mu = \lfloor v \rceil = \sum_j \lfloor v_j \rceil \cdot b_j \in R_2$ (where $b_j = \zeta^j$).

2. Homomorphically map coeffs v_j to "$\mathbb{Z}_q$-slots" of certain ring S_q:

$$\sum v_j \cdot b_j \in R_q \quad \longmapsto \quad \sum v_j \cdot c_j \in S_q.$$

(Change of basis, analogous to homomorphic DFT.)

3. Batch-round: homom'ly apply $\lfloor \cdot \rceil$ on all $\mathbb{Z}_q$-slots at once [SV'11]:

$$\sum v_j \cdot c_j \in S_q \quad \longmapsto \quad \sum \lfloor v_j \rceil \cdot c_j \in S_2.$$

4. Homomorphically reverse-map $\mathbb{Z}_2$-slots back to B-coeffs:

$$\sum \lfloor v_j \rceil \cdot c_j \in S_2 \quad \longmapsto \quad \sum \lfloor v_j \rceil \cdot b_j = \mu \in R_2.$$

(Akin to homomorphic DFT^{-1}.)

Algebra: Slots and CRT Sets

- Let $1 = \ell_0 | \ell_1 | \ell_2 | \cdots$ (all odd), and $S^{(i)} = \mathcal{O}_{\ell_i} = \mathbb{Z}[\zeta_{\ell_i}]$.
  Identifying $\zeta_{\ell_i}^{\ell_i/\ell_{i-1}} = \zeta_{\ell_{i-1}}$, we get a tower $S^{(i)}/S^{(i-1)}/\cdots/\mathbb{Z}$.

Algebra: Slots and CRT Sets

- Let $1 = \ell_0 | \ell_1 | \ell_2 | \cdots$ (all odd), and $S^{(i)} = \mathcal{O}_{\ell_i} = \mathbb{Z}[\zeta_{\ell_i}]$.

  Identifying $\zeta_{\ell_i}^{\ell_i/\ell_{i-1}} = \zeta_{\ell_{i-1}}$, we get a tower $S^{(i)}/S^{(i-1)}/\cdots/\mathbb{Z}$.
- In $S = S^{(i)}$, 2 factors into distinct prime ideals, like so:

$$
\begin{tikzcd}
S^{(2)} = \mathcal{O}_{91} \arrow[d, dash] \\
S^{(1)} = \mathcal{O}_7 \arrow[d, dash] \\
\mathbb{Z} = \mathcal{O}_1
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathfrak{p}_{1,1} \arrow[dr, dash] & \mathfrak{p}_{1,2} \arrow[d, dash] & \mathfrak{p}_{1,3} \arrow[dl, dash] & & \mathfrak{p}_{2,1} \arrow[dr, dash] & \mathfrak{p}_{2,2} \arrow[d, dash] & \mathfrak{p}_{2,3} \arrow[dl, dash] \\
& \mathfrak{p}_1 \arrow[drr, dash] & & & & \mathfrak{p}_2 \arrow[dll, dash] & \\
& & & 2 & & &
\end{tikzcd}
$$

Algebra: Slots and CRT Sets

- Let $1 = \ell_0 | \ell_1 | \ell_2 | \cdots$ (all odd), and $S^{(i)} = \mathcal{O}_{\ell_i} = \mathbb{Z}[\zeta_{\ell_i}]$.

  Identifying $\zeta_{\ell_i}^{\ell_i/\ell_{i-1}} = \zeta_{\ell_{i-1}}$, we get a tower $S^{(i)}/S^{(i-1)}/\cdots/\mathbb{Z}$.

- In $S = S^{(i)}$, 2 factors into distinct prime ideals, like so:

$$\begin{tikzcd}
S^{(2)} = \mathcal{O}_{91} \arrow[d, dash] & \mathfrak{p}_{1,1} \arrow[dr, dash] & \mathfrak{p}_{1,2} \arrow[d, dash] & \mathfrak{p}_{1,3} \arrow[dl, dash] & & \mathfrak{p}_{2,1} \arrow[dr, dash] & \mathfrak{p}_{2,2} \arrow[d, dash] & \mathfrak{p}_{2,3} \arrow[dl, dash] \\
S^{(1)} = \mathcal{O}_{7} \arrow[d, dash] & & \mathfrak{p}_1 \arrow[drr, dash] & & & & \mathfrak{p}_2 \arrow[dll, dash] & \\
\mathbb{Z} = \mathcal{O}_1 & & & & 2 & & &
\end{tikzcd}$$

- By Chinese Rem Thm, $S_2 \cong \bigoplus_j (S/\mathfrak{p}_j)$ via natural homomorphism.

Algebra: Slots and CRT Sets

- Let $1 = \ell_0 | \ell_1 | \ell_2 | \cdots$ (all odd), and $S^{(i)} = \mathcal{O}_{\ell_i} = \mathbb{Z}[\zeta_{\ell_i}]$.

  Identifying $\zeta_{\ell_i}^{\ell_i/\ell_{i-1}} = \zeta_{\ell_{i-1}}$, we get a tower $S^{(i)}/S^{(i-1)}/\cdots/\mathbb{Z}$.

- In $S = S^{(i)}$, 2 factors into distinct prime ideals, like so:

$$
\begin{tikzcd}
S^{(2)} = \mathcal{O}_{91} \arrow[d, dash] \\
S^{(1)} = \mathcal{O}_7 \arrow[d, dash] \\
\mathbb{Z} = \mathcal{O}_1
\end{tikzcd}
\qquad
\begin{tikzcd}[column sep=small]
\mathfrak{p}_{1,1} \arrow[dr, dash] & \mathfrak{p}_{1,2} \arrow[d, dash] & \mathfrak{p}_{1,3} \arrow[dl, dash] & & \mathfrak{p}_{2,1} \arrow[dr, dash] & \mathfrak{p}_{2,2} \arrow[d, dash] & \mathfrak{p}_{2,3} \arrow[dl, dash] \\
& \mathfrak{p}_1 \arrow[drr, dash] & & & & \mathfrak{p}_2 \arrow[dll, dash] & \\
& & & 2 & & &
\end{tikzcd}
$$

- By Chinese Rem Thm, $S_2 \cong \bigoplus_j (S/\mathfrak{p}_j)$ via natural homomorphism.

  "CRT set:" $C = \{c_j\} \subset S$ s.t. $c_j = 1 \pmod{\mathfrak{p}_j}$, $= 0 \pmod{\mathfrak{p}_{\neq j}}$.

  Mapping $v_j \in \mathbb{Z}_2 \mapsto v_j \cdot c_j \in S_2$ embeds $\mathbb{Z}_2$ into jth "slot" of S_2.

Algebra: Slots and CRT Sets

- Let $1 = \ell_0 | \ell_1 | \ell_2 | \cdots$ (all odd), and $S^{(i)} = \mathcal{O}_{\ell_i} = \mathbb{Z}[\zeta_{\ell_i}]$.

  Identifying $\zeta_{\ell_i}^{\ell_i/\ell_{i-1}} = \zeta_{\ell_{i-1}}$, we get a tower $S^{(i)}/S^{(i-1)}/\cdots/\mathbb{Z}$.

- In $S = S^{(i)}$, 2 factors into distinct prime ideals, like so:

$$\begin{tikzcd}
S^{(2)} = \mathcal{O}_{91} \arrow[d, dash] \\
S^{(1)} = \mathcal{O}_{7} \arrow[d, dash] \\
\mathbb{Z} = \mathcal{O}_1
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathfrak{p}_{1,1} \arrow[dr, dash] & \mathfrak{p}_{1,2} \arrow[d, dash] & \mathfrak{p}_{1,3} \arrow[dl, dash] & & \mathfrak{p}_{2,1} \arrow[dr, dash] & \mathfrak{p}_{2,2} \arrow[d, dash] & \mathfrak{p}_{2,3} \arrow[dl, dash] \\
& \mathfrak{p}_1 \arrow[drr, dash] & & & & \mathfrak{p}_2 \arrow[dll, dash] & \\
& & & 2 & & &
\end{tikzcd}$$

- By Chinese Rem Thm, $S_2 \cong \bigoplus_j (S/\mathfrak{p}_j)$ via natural homomorphism.

  "CRT set:" $C = \{c_j\} \subset S$ s.t. $c_j = 1 \pmod{\mathfrak{p}_j}$, $= 0 \pmod{\mathfrak{p}_{\neq j}}$.

  Mapping $v_j \in \mathbb{Z}_2 \mapsto v_j \cdot c_j \in S_2$ embeds $\mathbb{Z}_2$ into jth "slot" of S_2.

- Can factor $C_i = C_i' \cdot C_{i-1}$: let $c_k' = 1 \pmod{\mathfrak{p}_{\star,k}}$, $= 0 \pmod{\mathfrak{p}_{\star,\neq k}}$.

Algebra: Slots and CRT Sets

- Let $1 = \ell_0 | \ell_1 | \ell_2 | \cdots$ (all odd), and $S^{(i)} = \mathcal{O}_{\ell_i} = \mathbb{Z}[\zeta_{\ell_i}]$.
  Identifying $\zeta_{\ell_i}^{\ell_i/\ell_{i-1}} = \zeta_{\ell_{i-1}}$, we get a tower $S^{(i)}/S^{(i-1)}/\cdots/\mathbb{Z}$.
- In $S = S^{(i)}$, 2 factors into distinct prime ideals, like so:

$$
\begin{array}{c}
S^{(2)} = \mathcal{O}_{91} \\
| \\
S^{(1)} = \mathcal{O}_7 \\
| \\
\mathbb{Z} = \mathcal{O}_1
\end{array}
\qquad
\begin{tikzcd}[arrows=dash]
\mathfrak{p}_{1,1} \arrow[dr] & \mathfrak{p}_{1,2} \arrow[d] & \mathfrak{p}_{1,3} \arrow[dl] & & \mathfrak{p}_{2,1} \arrow[dr] & \mathfrak{p}_{2,2} \arrow[d] & \mathfrak{p}_{2,3} \arrow[dl] \\
& \mathfrak{p}_1 \arrow[drr] & & & & \mathfrak{p}_2 \arrow[dll] & \\
& & & 2 & & &
\end{tikzcd}
$$

- By Chinese Rem Thm, $S_2 \cong \bigoplus_j (S/\mathfrak{p}_j)$ via natural homomorphism.
  "CRT set:" $C = \{c_j\} \subset S$ s.t. $c_j = 1 \pmod{\mathfrak{p}_j}$, $= 0 \pmod{\mathfrak{p}_{\neq j}}$.
  Mapping $v_j \in \mathbb{Z}_2 \mapsto v_j \cdot c_j \in S_2$ embeds $\mathbb{Z}_2$ into jth "slot" of S_2.
- Can factor $C_i = C_i' \cdot C_{i-1}$: let $c_k' = 1 \pmod{\mathfrak{p}_{\star,k}}$, $= 0 \pmod{\mathfrak{p}_{\star,\neq k}}$.
- Similarly for $S_q \cong \bigoplus_j (S/\mathfrak{p}_j^{\lg q})$.

Mapping Coeffs to Slots: Overview

- Choose S so that S_q has $\geq \deg(R/\mathbb{Z})$ $\mathbb{Z}_q$-slots, via:

$$(v_j) \in \mathbb{Z}_q^{k/2} \longmapsto \sum v_j \cdot c_j \bmod q$$

for an appropriate CRT set $C = \{c_j\} \subset S$ of size $k/2$.

Mapping Coeffs to Slots: Overview

- Choose S so that S_q has $\geq \deg(R/\mathbb{Z})$ $\mathbb{Z}_q$-slots, via:

$$(v_j) \in \mathbb{Z}_q^{k/2} \;\longmapsto\; \sum v_j \cdot c_j \bmod q$$

  for an appropriate CRT set $C = \{c_j\} \subset S$ of size $k/2$.

- Our goal: homomorphically map $\sum v_j \cdot b_j \in R_q \;\longmapsto\; \sum v_j \cdot c_j \in S_q$.

Mapping Coeffs to Slots: Overview

- Choose S so that S_q has $\geq \deg(R/\mathbb{Z})$ $\mathbb{Z}_q$-slots, via:

$$(v_j) \in \mathbb{Z}_q^{k/2} \longmapsto \sum v_j \cdot c_j \bmod q$$

  for an appropriate CRT set $C = \{c_j\} \subset S$ of size $k/2$.

- Our goal: homomorphically map $\sum v_j \cdot b_j \in R_q \longmapsto \sum v_j \cdot c_j \in S_q$.

  Equivalently, evaluate the $\mathbb{Z}$-linear* map $L \colon R \to S$ defined by

$$L(b_j) = c_j.$$

*$\mathbb{Z}$-linear: $L(b + b') = L(b) + L(b')$, $L(v \cdot b) = v \cdot L(b)$ for any $b, b' \in R, v \in \mathbb{Z}$.

Mapping Coeffs to Slots: Overview

- Choose S so that S_q has $\geq \deg(R/\mathbb{Z})$ $\mathbb{Z}_q$-slots, via:

  $$(v_j) \in \mathbb{Z}_q^{k/2} \longmapsto \sum v_j \cdot c_j \bmod q$$

  for an appropriate CRT set $C = \{c_j\} \subset S$ of size $k/2$.

- Our goal: homomorphically map $\sum v_j \cdot b_j \in R_q \longmapsto \sum v_j \cdot c_j \in S_q$.

  Equivalently, evaluate the $\mathbb{Z}$-linear* map $L \colon R \to S$ defined by

  $$L(b_j) = c_j.$$

- Ring-switching [GHPS'12] lets us eval any R'-linear map $L \colon R \to R'$

*$\mathbb{Z}$-linear: $L(b + b') = L(b) + L(b')$, $L(v \cdot b) = v \cdot L(b)$ for any $b, b' \in R, v \in \mathbb{Z}$.

Mapping Coeffs to Slots: Overview

- Choose S so that S_q has $\geq \deg(R/\mathbb{Z})$ $\mathbb{Z}_q$-slots, via:

$$(v_j) \in \mathbb{Z}_q^{k/2} \longmapsto \sum v_j \cdot c_j \bmod q$$

  for an appropriate CRT set $C = \{c_j\} \subset S$ of size $k/2$.

- Our goal: homomorphically map $\sum v_j \cdot b_j \in R_q \longmapsto \sum v_j \cdot c_j \in S_q$.

  Equivalently, evaluate the $\mathbb{Z}$-linear* map $L \colon R \to S$ defined by

$$L(b_j) = c_j.$$

- Ring-switching [GHPS'12] lets us eval any R'-linear map $L \colon R \to R'$

  ...but only for a subring $R' \subseteq R$.

*$\mathbb{Z}$-linear: $L(b + b') = L(b) + L(b')$, $L(v \cdot b) = v \cdot L(b)$ for any $b, b' \in R, v \in \mathbb{Z}$.

Mapping Coeffs to Slots: Overview

- Choose S so that S_q has $\geq \deg(R/\mathbb{Z})$ $\mathbb{Z}_q$-slots, via:

$$(v_j) \in \mathbb{Z}_q^{k/2} \longmapsto \sum v_j \cdot c_j \bmod q$$

  for an appropriate CRT set $C = \{c_j\} \subset S$ of size $k/2$.

- Our goal: homomorphically map $\sum v_j \cdot b_j \in R_q \longmapsto \sum v_j \cdot c_j \in S_q$.

  Equivalently, evaluate the $\mathbb{Z}$-linear* map $L\colon R \to S$ defined by

$$L(b_j) = c_j.$$

- Ring-switching [GHPS'12] lets us eval any R'-linear map $L\colon R \to R'$

  ...but only for a subring $R' \subseteq R$.

Goal for Remainder of Talk

- Extend ring-switching to (efficiently) handle $\mathbb{Z}$-linear maps $L\colon R \to S$.

*$\mathbb{Z}$-linear: $L(b + b') = L(b) + L(b')$, $L(v \cdot b) = v \cdot L(b)$ for any $b, b' \in R, v \in \mathbb{Z}$.

Algebra: Combining Cyclotomic Rings

- Let $R = \mathcal{O}_k$, $S = \mathcal{O}_\ell$. Let $d = \gcd(k, \ell)$ and $m = \mathrm{lcm}(k, \ell)$.

Algebra: Combining Cyclotomic Rings

- Let $R = \mathcal{O}_k$, $S = \mathcal{O}_\ell$. Let $d = \gcd(k, \ell)$ and $m = \operatorname{lcm}(k, \ell)$.

$$\begin{tikzcd}[arrows=dash]
 & T = R + S = \mathcal{O}_m \text{ ("compositum")} \arrow[dl] \arrow[dr] & \\
R \arrow[dr] & & S \arrow[dl] \\
 & E = R \cap S = \mathcal{O}_d &
\end{tikzcd}$$

Algebra: Combining Cyclotomic Rings

- Let $R = \mathcal{O}_k$, $S = \mathcal{O}_\ell$. Let $d = \gcd(k, \ell)$ and $m = \mathrm{lcm}(k, \ell)$.

$$\begin{tikzcd}
 & T = R + S = \mathcal{O}_m \quad (\text{“compositum”}) \arrow[dl, dash] \arrow[dr, dash] & \\
R \arrow[dr, dash] & & S \arrow[dl, dash] \\
 & E = R \cap S = \mathcal{O}_d &
\end{tikzcd}$$

- Compositum T as a tensor product of R, S, where $\otimes$ is E-bilinear:

$$T \cong (R/E) \otimes (S/E) := \left\{ \sum e_{i,j}(r_i \otimes s_j) : e_{i,j} \in E, r_i \in R, s_j \in S \right\}.$$

Algebra: Combining Cyclotomic Rings

- Let $R = \mathcal{O}_k$, $S = \mathcal{O}_\ell$. Let $d = \gcd(k, \ell)$ and $m = \operatorname{lcm}(k, \ell)$.

$$\begin{tikzcd}[column sep=small]
& T = R + S = \mathcal{O}_m \quad (\text{“compositum”}) \arrow[dl, dash] \arrow[dr, dash] & \\
R \arrow[dr, dash] & & S \arrow[dl, dash] \\
& E = R \cap S = \mathcal{O}_d &
\end{tikzcd}$$

- Compositum T as a tensor product of R, S, where $\otimes$ is E-bilinear:

$$T \cong (R/E) \otimes (S/E) := \left\{ \sum e_{i,j}(r_i \otimes s_j) : e_{i,j} \in E, r_i \in R, s_j \in S \right\}.$$

Easy Lemma

- For any E-linear $L \colon R \to S$, there is an S-linear $\bar{L} \colon T \to S$ that agrees with L on R.

Algebra: Combining Cyclotomic Rings

- Let $R = \mathcal{O}_k$, $S = \mathcal{O}_\ell$. Let $d = \gcd(k, \ell)$ and $m = \mathrm{lcm}(k, \ell)$.

$$\begin{tikzcd}[column sep=small]
 & T = R + S = \mathcal{O}_m \text{ (“compositum”)} \arrow[dl, dash] \arrow[dr, dash] & \\
R \arrow[dr, dash] & & S \arrow[dl, dash] \\
 & E = R \cap S = \mathcal{O}_d &
\end{tikzcd}$$

- Compositum T as a tensor product of R, S, where $\otimes$ is E-bilinear:

$$T \cong (R/E) \otimes (S/E) := \left\{ \sum e_{i,j}(r_i \otimes s_j) : e_{i,j} \in E, r_i \in R, s_j \in S \right\}.$$

Easy Lemma

- For any E-linear $L\colon R \to S$, there is an S-linear $\bar{L}\colon T \to S$ that agrees with L on R.
- Proof: define $\bar{L}$ by $\bar{L}(r \otimes s) = L(r) \cdot s \in S$.

Enhanced Ring-Switching: First Attempt

- Let $R = \mathcal{O}_k$, $S = \mathcal{O}_\ell$ be s.t. $\gcd(k, \ell) = 1$, $\operatorname{lcm}(k, \ell) = k\ell$.

Enhanced Ring-Switching: First Attempt

- Let $R = \mathcal{O}_k$, $S = \mathcal{O}_\ell$ be s.t. $\gcd(k, \ell) = 1$, $\operatorname{lcm}(k, \ell) = k\ell$.

$$
\begin{tikzcd}
 & T = \mathcal{O}_{k\ell} \arrow[dr, "\bar{L}"] & \\
R \arrow[ur, "\text{embed}"] \arrow[rr, dashed, "L", "\text{(induced)}"'] \arrow[dr, dash] & & S \\
 & E = \mathbb{Z} \arrow[ur, dash] &
\end{tikzcd}
$$

Enhanced Ring-Switching: First Attempt

- Let $R = \mathcal{O}_k$, $S = \mathcal{O}_\ell$ be s.t. $\gcd(k, \ell) = 1$, $\operatorname{lcm}(k, \ell) = k\ell$.

$$
\begin{tikzcd}
 & T = \mathcal{O}_{k\ell} \arrow[dr, "\bar{L}"] & \\
R \arrow[ur, "\text{embed}"] \arrow[rr, dashed, "L", "\text{(induced)}"'] \arrow[dr, dash] & & S \arrow[dl, dash] \\
 & E = \mathbb{Z} &
\end{tikzcd}
$$

- To homom'ly eval. $\mathbb{Z}$-linear $L \colon R \to S$ on an encryption of $v \in R_q$,

Enhanced Ring-Switching: First Attempt

- Let $R = \mathcal{O}_k$, $S = \mathcal{O}_\ell$ be s.t. $\gcd(k, \ell) = 1$, $\mathrm{lcm}(k, \ell) = k\ell$.

$$
\begin{tikzcd}
& T = \mathcal{O}_{k\ell} \arrow[dr, "\bar{L}"] & \\
R \arrow[ur, "\text{embed}"] \arrow[rr, dashed, "L", "\text{(induced)}"'] \arrow[dr, dash] & & S \arrow[dl, dash] \\
& E = \mathbb{Z} &
\end{tikzcd}
$$

- To homom'ly eval. $\mathbb{Z}$-linear $L \colon R \to S$ on an encryption of $v \in R_q$,
  1. Trivially embed ciphertext $R \to T$ (still encrypts v).
  2. Homomorphically apply S-linear $\bar{L} \colon T \to S$ using ring-switching.
  - ✔ We now have an encryption of $\bar{L}(v) = L(v)$!

Enhanced Ring-Switching: First Attempt

- Let $R = \mathcal{O}_k$, $S = \mathcal{O}_\ell$ be s.t. $\gcd(k, \ell) = 1$, $\mathrm{lcm}(k, \ell) = k\ell$.

$$\begin{tikzcd} & T = \mathcal{O}_{k\ell} \arrow[dr, "\bar{L}"] & \\ R \arrow[ur, "\text{embed}"] \arrow[rr, dashed, "L", "\text{(induced)}"'] \arrow[dr, dash] & & S \\ & E = \mathbb{Z} \arrow[ur, dash] & \end{tikzcd}$$

- To homom'ly eval. $\mathbb{Z}$-linear $L \colon R \to S$ on an encryption of $v \in R_q$,
  1. Trivially embed ciphertext $R \to T$ (still encrypts v).
  2. Homomorphically apply S-linear $\bar{L} \colon T \to S$ using ring-switching.
  - ✔ We now have an encryption of $\bar{L}(v) = L(v)$!

✗✗ Problem: degree of T is quadratic, therefore so is runtime & space.

Enhanced Ring-Switching: First Attempt

- Let $R = \mathcal{O}_k$, $S = \mathcal{O}_\ell$ be s.t. $\gcd(k, \ell) = 1$, $\mathrm{lcm}(k, \ell) = k\ell$.

$$
\begin{tikzcd}
 & T = \mathcal{O}_{k\ell} \arrow[dr, "\bar{L}"] & \\
R \arrow[ur, "\text{embed}"] \arrow[rr, dashed, "L", "\text{(induced)}"'] \arrow[dr, dash] & & S \\
 & E = \mathbb{Z} \arrow[ur, dash] &
\end{tikzcd}
$$

- To homom'ly eval. $\mathbb{Z}$-linear $L \colon R \to S$ on an encryption of $v \in R_q$,
  1. Trivially embed ciphertext $R \to T$ (still encrypts v).
  2. Homomorphically apply S-linear $\bar{L} \colon T \to S$ using ring-switching.

  ✔ We now have an encryption of $\bar{L}(v) = L(v)$!

✗✗ Problem: degree of T is quadratic, therefore so is runtime & space. This is inherent if we treat L as a generic $\mathbb{Z}$-linear map!

Enhanced Ring-Switching, Efficiently

Key Ideas

- The $\mathbb{Z}$-linear $L \colon R \to S$ given by $L(B) = C$ is "highly structured," because B, C are product sets.

Enhanced Ring-Switching, Efficiently

Key Ideas

- The $\mathbb{Z}$-linear $L\colon R \to S$ given by $L(B) = C$ is "highly structured," because B, C are product sets.
- Gradually map B to C through a sequence of "hybrid rings" $H^{(i)}$, via $E^{(i)}$-linear functions that each send a factor of B to one of C.

$$
\begin{tikzcd}
 & T^{(1)} \arrow[dr] & & T^{(2)} \arrow[dr] & \\
B \subset R = H^{(0)} \arrow[ur, "\text{embed}"] \arrow[rr, dashed, "E^{(1)}\text{-linear}", "\text{(induced)}"'] \arrow[dr, dash] & & H^{(1)} \arrow[ur, "\text{embed}"] \arrow[rr, dashed, "E^{(2)}\text{-linear}", "\text{(induced)}"'] \arrow[dr, dash] & & H^{(2)} = S \supset C \\
 & E^{(1)} \arrow[ur, dash] & & E^{(2)} \arrow[ur, dash] &
\end{tikzcd}
$$

Enhanced Ring-Switching, Efficiently

Key Ideas

- The $\mathbb{Z}$-linear $L\colon R \to S$ given by $L(B) = C$ is "highly structured," because B, C are product sets.
- Gradually map B to C through a sequence of "hybrid rings" $H^{(i)}$, via $E^{(i)}$-linear functions that each send a factor of B to one of C.
- Ensure small compositums $T^{(i)} = H^{(i-1)} + H^{(i)}$ via large gcd's: replace prime factors of k with those of ℓ, one at a time.

$$\begin{tikzcd}
 & T^{(1)} \arrow[dr] & & T^{(2)} \arrow[dr] & \\
B \subset R = H^{(0)} \arrow[ur, "\text{embed}"] \arrow[rr, dashed, "E^{(1)}\text{-linear}", "\text{(induced)}"'] \arrow[dr, dash] & & H^{(1)} \arrow[ur, "\text{embed}"] \arrow[rr, dashed, "E^{(2)}\text{-linear}", "\text{(induced)}"'] \arrow[dr, dash] & & H^{(2)} = S \supset C \\
 & E^{(1)} \arrow[ur, dash] & & E^{(2)} \arrow[ur, dash] &
\end{tikzcd}$$

Toy Example

- $R = \mathcal{O}_8$, basis $B = B'_8 \cdot B'_4 = \{1, \zeta_8\} \cdot \{1, \zeta_4\}$.

Toy Example

- $R = \mathcal{O}_8$, basis $B = B'_8 \cdot B'_4 = \{1, \zeta_8\} \cdot \{1, \zeta_4\}$.
- $S = \mathcal{O}_{7 \cdot 13}$, CRT set $C = C'_7 \cdot C'_{91} = \{c_1, c_2\} \cdot \{c'_1, c'_2, c'_3\}$.

Toy Example

- $R = \mathcal{O}_8$, basis $B = B_8' \cdot B_4' = \{1, \zeta_8\} \cdot \{1, \zeta_4\}$.
- $S = \mathcal{O}_{7\cdot 13}$, CRT set $C = C_7' \cdot C_{91}' = \{c_1, c_2\} \cdot \{c_1', c_2', c_3'\}$.

$$
\begin{tikzcd}
\begin{array}{c} B_8' \cdot B_4' \\ \subset \mathcal{O}_8 \end{array} \arrow[rr, dashed, "B_8' \to C_7'", "\text{fix } B_4'"'] \arrow[dr, dash] & & \begin{array}{c} B_4' \cdot C_7' \\ \subset \mathcal{O}_{4\cdot 7} \end{array} \arrow[rr, dashed, "B_4' \to C_{91}'", "\text{fix } C_7'"'] \arrow[dl, dash] \arrow[dr, dash] & & \begin{array}{c} C_7' \cdot C_{91}' \\ \subset \mathcal{O}_{7\cdot 13} \end{array} \arrow[dl, dash] \\
& \mathcal{O}_4 & & \mathcal{O}_7 &
\end{tikzcd}
$$

Toy Example

- $R = \mathcal{O}_8$, basis $B = B'_8 \cdot B'_4 = \{1, \zeta_8\} \cdot \{1, \zeta_4\}$.
- $S = \mathcal{O}_{7\cdot 13}$, CRT set $C = C'_7 \cdot C'_{91} = \{c_1, c_2\} \cdot \{c'_1, c'_2, c'_3\}$.

$$
\begin{tikzcd}
{B'_8 \cdot B'_4 \subset \mathcal{O}_8} \arrow[rr, dashed, "B'_8 \to C'_7", "\text{fix } B'_4"'] \arrow[dr, dash] & & {B'_4 \cdot C'_7 \subset \mathcal{O}_{4\cdot 7}} \arrow[rr, dashed, "B'_4 \to C'_{91}", "\text{fix } C'_7"'] \arrow[dl, dash] \arrow[dr, dash] & & {C'_7 \cdot C'_{91} \subset \mathcal{O}_{7\cdot 13}} \arrow[dl, dash] \\
& \mathcal{O}_4 & & \mathcal{O}_7 &
\end{tikzcd}
$$

- In general, switch through $\leq \log(\deg(R/\mathbb{Z})) = \log(\lambda)$ hybrid rings, one for each prime factor of k.

Final Thoughts

- Gradually converting B to C via hybrid rings is roughly analogous to a log-depth FFT butterfly network.

Final Thoughts

- Gradually converting B to C via hybrid rings is roughly analogous to a log-depth FFT butterfly network.
- Technique should also be useful for homomorphically evaluating other signal-processing transforms having "sparse decompositions."

Final Thoughts

- Gradually converting B to C via hybrid rings is roughly analogous to a log-depth FFT butterfly network.
- Technique should also be useful for homomorphically evaluating other signal-processing transforms having "sparse decompositions."
- Practical implementation and evaluation are underway.

Final Thoughts

- Gradually converting B to C via hybrid rings is roughly analogous to a log-depth FFT butterfly network.
- Technique should also be useful for homomorphically evaluating other signal-processing transforms having "sparse decompositions."
- Practical implementation and evaluation are underway.

Thanks!