Fully Homomorphic Encryption and Bootstrapping

Craig Gentry and Shai Halevi

Fully Homomorphic Encryption (FHE)

- A FHE scheme can evaluate unbounded depth circuits
  - Not limited by bound specified at Setup
  - Parameters (like size of ciphertext) do not depend on evaluated depth
- So far, GSW scheme can evaluate only depth $\log_{N+1} q$
  - How do we make it *fully* homomorphic?

- Bootstrapping: A way to get FHE…

Self-Referential Encrypted Computation

A Digression into Philosophy…

- Can the human mind understand itself?
  - Or, as a mind becomes more complex, does the task of understanding also become more complex, so that self-understanding it always just out of reach?

- Self-reference often causes problems, even in mathematics and CS
  - Godel's incompleteness theorem
  - Turing's Halting Problem

Philosophy Meets Cryptography

- Can a homomorphic encryption scheme decrypt itself?
  - We can try to plug the decryption function Dec(·,·) into Eval.
  - If we run $\mathrm{Eval}_{pk}(\mathrm{Dec}(\cdot,\cdot), c_1, \ldots, c_t)$, does it work?
  - Suppose our HE scheme can Eval depth-d circuits:
    - Is it always true that HE's Dec function has depth > d?
    - Is Dec(·,·) always just beyond the Eval capacity of the HE scheme?

- Bootstrapping = the process of running Eval on Dec(·,·).

Bootstrapping: Assuming we can do it, why is it useful?

Bootstrapping: Refreshing a Ciphertext

- So far, we can evaluate bounded-depth circuits f:

μ_1

μ_2

…

μ_t

$f(\mu_1, \mu_2, \ldots, \mu_t)$

We have a noisy evaluated ciphertext y

We want to get another y with less noise

Bootstrapping *refreshes* ciphertexts, using the *encrypted secret key*.

Bootstrapping: Refreshing a Ciphertext

- For ciphertext c, consider the function $D_c(\cdot) = \mathrm{Dec}(\cdot, c)$
- Suppose we can Eval depth d, but $D_c(\cdot)$ has depth d-1.
- Include in the public key also $\mathrm{Enc}_{pk}(sk)$

c y

New encryption of y, with *less noise.*

$c' =$ $D_c(sk)$ $= \mathrm{Dec}(sk,c) = y$

sk_1

sk_2

…

sk_n

Homomorphic computation applied only to the “fresh” encryption of *sk*.

Bootstrapping Theorem (Informal)

- Suppose $\mathcal{E}$ is a HE scheme
  - that can evaluate arithmetic circuits of depth d
  - whose decryption algorithm is a circuit of depth d-1
- Call $\mathcal{E}$ a "bootstrappable" HE scheme

- Thm: From a bootstrappable somewhat homomorphic scheme, we can construct a fully homomorphic scheme.

- Technique: Refresh noisy ciphertexts by evaluating the decryption circuit homomorphically

Bootstrapping: Can we do it?

Let's Look at the Decryption Circuit…

- Typically in LWE-based encryption schemes, if c encrypts μ under secret key vector s, then:

$$\mu = [[\langle \mathbf{c}, \mathbf{t} \rangle]_q]_2$$

where $[\cdot]_q$ denotes reduction modulo q into the range (-q/2,q/2].

Decryption in GSW

- GSW fits the template: ($\mu = [[\langle \mathbf{c}, \mathbf{t} \rangle]_q]_2$)
  - $\mathbf{C} \cdot \mathbf{v} = \mu \cdot \mathbf{v} + 2 \cdot \mathbf{e} \bmod q$
  - $\langle \mathbf{c}, \mathbf{v} \rangle = \mu + 2 \cdot e \bmod q$
  - $\langle \mathsf{BitDecomp}^{-1}(\mathbf{c}), \mathbf{t} \rangle = \mu + 2 \cdot e \bmod q$
  - $[[\langle \mathsf{BitDecomp}^{-1}(\mathbf{c}), \mathbf{t} \rangle]_q]_2 = \mu$

How Complex Is Decryption?

$$\mu = [[\langle \mathbf{c}, \mathbf{t} \rangle]_q]_2$$

- If q is polynomial (in the security parameter λ) then decryption is in NC1 (log-depth circuits).
  - But wait – isn't q really large?
  - q depends on the Eval capacity of the scheme
  - Ideally, we would like the complexity of Dec to be independent of the Eval capacity.

Modulus Reduction Magic Trick

- Suppose c encrypts μ – that is, $\mu = [[\langle c,t\rangle]_q]_2$.
- Let's pick $p<q$ and set $c^* = (p/q)¢c$, rounded.
- Crazy idea: Maybe it is true that:
  - c^* encrypts μ : $\mu = [[\langle c^*,t\rangle]_p]_2$ (new inner modulus).
- Surprisingly, this works!

- After modulus reduction (and dimension reduction), the size of the ciphertext is independent of the complexity of the function that was evaluated!!

Modulus Reduction Magic Trick, Details

<u>Scaling lemma</u>: Let $p<q$ be odd moduli. Suppose $\mu = [[\langle c,t\rangle]_q]_2$ and $|[\langle c,t\rangle]_q| < q/2 - (q/p)\cdot l_1(t)$. Set $c' = (p/q)c$ and set c'' to be the integer vector closest to c' such that $c'' = c \bmod 2$. Then $\mu = [[\langle c'',t\rangle]_p]_2$.

<u>Annotated Proof</u>:

1. For some k, $[\langle c,t\rangle]_q = \langle c,t\rangle - kq$.
2. $(p/q)|[\langle c,t\rangle]_q| = \langle c',t\rangle - kp$.
3. $|\langle c''-c',t\rangle| < l_1(t)$.
4. Thus, $|\langle c'',t\rangle - kp| < (p/q)\ |[\langle c,t\rangle]_q| + l_1(t) < p/2$.
5. So, $[\langle c'',t\rangle]_p = \langle c'',t\rangle - kp$.
6. Since $c'' = c \bmod 2$ and $p = q \bmod 2$, we get $[\langle c'',t\rangle]_p]_2 = [\langle c,t\rangle]_q]_2$.

1. Imagine $\langle c,t\rangle$ is close to kq.
2. Then $\langle c',t\rangle$ is close to kp.
3. $\langle c'',t\rangle$ also close to kp if s small.

Modulus Reduction Magic Trick, Notes

- [ACPS 2009] proved LWE hard even if t is <u>small</u>:
  - t chosen from the same distribution as the noise e
    - With coefficients of size poly in the security parameter.
  - For t of polynomial size, we can modulus reduce to a modulus p of polynomial size, before bootstrapping.

- Bottom Line: After some processing, decryption for LWE-based encryption schemes (like GSW) is in NC1.
  - Complexity of Dec is independent of Eval capacity.

Evaluating NC1 Circuits in GSW

- Naïve way: Just to log levels of NAND
- Each level multiplies noise by polynomial factor.

$$\begin{aligned} \mathbf{C}^{\mathrm{NAND}} \cdot \mathbf{v} &= (\mathbf{I} - \mathbf{C_1} \cdot \mathbf{C_2}) \cdot \mathbf{v} \\ &= (1 - \mu_1 \cdot \mu_2) \cdot \mathbf{v} - (\mu_2 \cdot \mathbf{e_1} + \mathbf{C_1} \cdot \mathbf{e_2}) \end{aligned}$$

- Log levels multiplies noise by quasi-polynomial factor.
- Bad consequence = weak security: Based on LWE for quasi-polynomial approximation factors.

Part II: Bootstrapping and Barrington's Theorem

Focusing on Brakerski and Vaikuntanathan's method to bootstrap the Gentry-Sahai-Waters scheme

Better Way to Evaluate NC1 Circuits?

- Goal: Base security of FHE on LWE with poly factors.
  - Evaluate NC1 circuits in a more "noise-friendly" way so that there is only polynomial noise blowup.

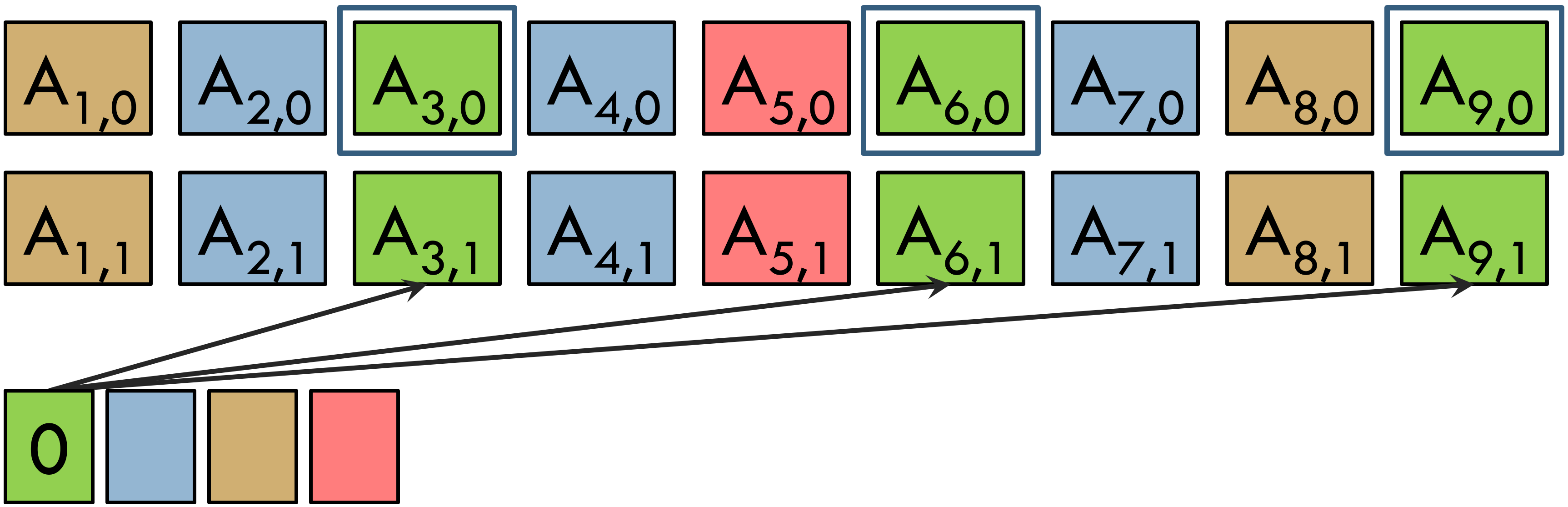
- Barrington's Theorem
  - If f is computable by a d-depth Boolean circuit, then it is computable by a width-5 permutation branching program of length 4^d.
  - Corollary: every function in NC1 has a polynomial-length BP.

Width-5 Permutation Branching Programs

- BP for function f:
  - Consists of labeled permutations in the permutation group S_5 (which we represent as 5x5 permutation matrices)
  - S_5 is a non-abelian group: maybe $ab \neq ba$.

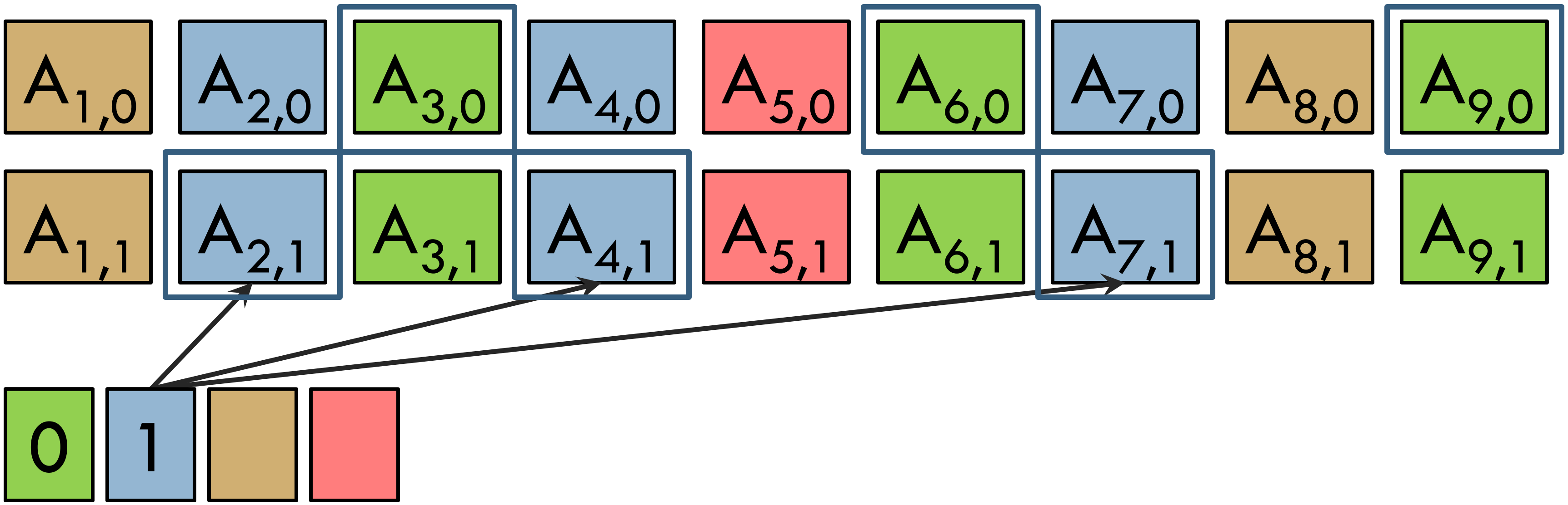
- To evaluate BP (hence f) on input X:
  - Map X to a subset S_X of the matrices (using labels)
  - Compute product of the matrices in S_X
  - Output 1 if the product is the identity matrix, 0 otherwise

Width-5 Permutation Branching Programs



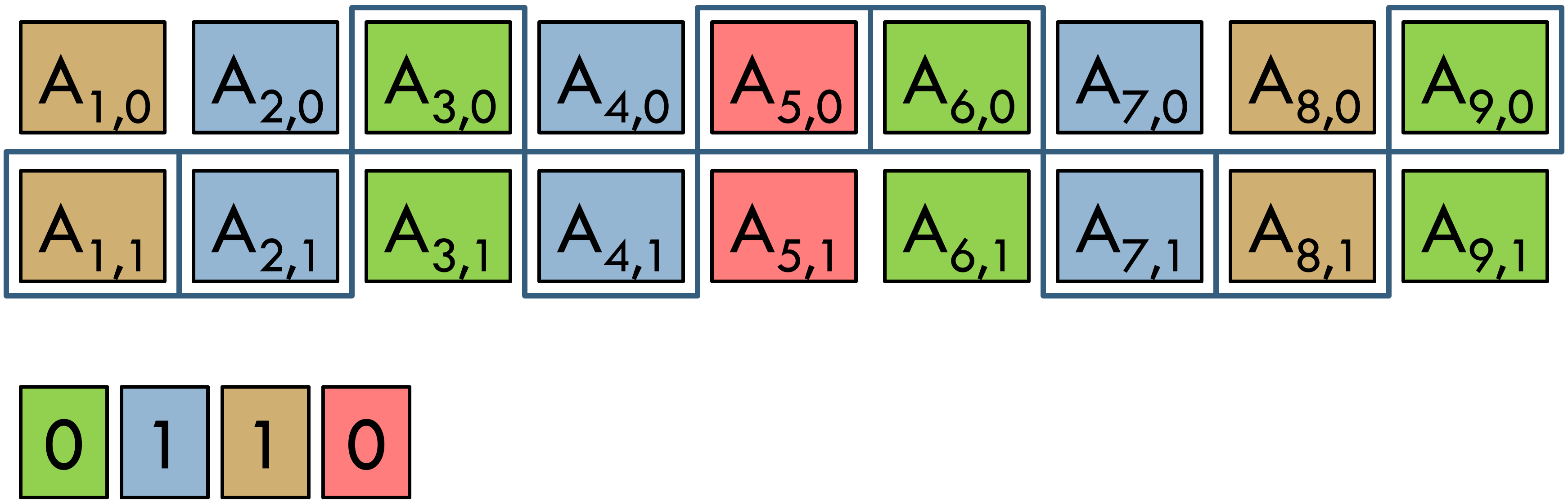
- Each $A_{i,b}$ is a 5x5 permutation matrix.
- This BP takes 4-bit inputs and has length 9

Width-5 Permutation Branching Programs



- Each $A_{i,b}$ is a 5x5 permutation matrix.
- This BP takes 4-bit inputs and has length 9

Width-5 Permutation Branching Programs



- Each $A_{i,b}$ is a 5x5 permutation matrix.
- This BP takes 4-bit inputs and has length 9
- Multiply the chosen 9 matrices together
  - If product is I, output 1. Otherwise, output 0.

Brakerski and Vaikuntanathan's Insight

- Multiplications in GSW increase noise asymmetrically.
- Moreover, this asymmetry is useful.
  - Can exploit it to evaluate permutation BPs with surprisingly little noise growth.

Warm Up: High Fan-in AND Gates

$$\mathbf{C_1} \cdot \mathbf{C_2} \cdot \mathbf{v} = \mu_1 \cdot \mu_2 \cdot \mathbf{v} + (\mu_2 \cdot \mathbf{e_1} + \mathbf{C_1} \cdot \mathbf{e_2})$$

- Binary Tree approach: AND t ciphertexts using a (log t)-depth binary tree.
  - Noise grows by $(N+1)^{\log t}$ factor.

- Left-to-right approach: AND t ciphertexts by multiplying sequentially from left to right
  - The i-th multiplication only **adds** $C_i' \cdot e_{i+1}$ to the error.
    - $C_i' \in \{0,1\}^{N \times N}$ is the aggregate-so-far
    - e_{i+1} is the (small) error of the (i+1)-th ciphertext.
  - Noise grows by $t(N+1)$ factor.

- Right-to-left approach: horrible!

Multiplying Permutation Matrices

$$\mathbf{C_1} \cdot \mathbf{C_2} \cdot \mathbf{v} = \mu_1 \cdot \mu_2 \cdot \mathbf{v} + (\mu_2 \cdot \mathbf{e_1} + \mathbf{C_1} \cdot \mathbf{e_2})$$

- Given kxk permutation matrices encrypted entry-wise, multiplying them left-to-right is best.
- Multiplying in the (i+1)-th permutation matrix **adds** about k(N+1) times the error of fresh ciphertexts.
- Essential fact used in analysis: In a permutation matrix, only one entry per column is nonzero.

Lattice-Based FHE as Secure as PKE [BV14]

- Bottom line:
  - GSW decryption can be computed homomorphically while increasing noise by a poly factor.
  - FHE can be based on LWE with poly approx factors.
    - The exponent can be made ε-close to that of current LWE-based PKE schemes.

Part IV: FHE from Non-Abelian Groups?

A somewhat promising framework for FHE inspired by Barrington's Theorem

Goal: Totally Different Approach to FHE

- FHE without noise?
  - Might also make (expensive) bootstrapping unnecessary

- How about FHE based on non-abelian groups?
  - Might avoid linear algebra attacks for ring-based schemes
  - Another chance to apply Barrington. ☺
  - Framework investigated by Nuida
    - ePrint 2014/07: "A Simple Framework for Noise-Free Construction of Fully Homomorphic Encryption from a Special Class of Non-commutative Groups"

Perfect Group Pairs

Groups (G, H) such that:

- H is a (proper, nontrivial) normal subgroup of G
  - $H = \{ghg^{-1} : g \in G, h \in H\}$
- G and H are perfect groups
  - Commutator subgroup $[G,G] = \langle g_1 g_2 g_1^{-1} g_2^{-1} : g_1, g_2 \in G \rangle$
  - G is "perfect" when $G = [G,G]$

Efficient Group Operations

- Randomization: Given a group (say, G) represented by some generators, output ≤n "random" G-elements that generate the group.

Hardness Assumption

- Subgroup Decision Assumption (for perfect group pairs): Given $\leq n$ elements that generate either G or H, hard to distinguish which.

FHE Construction

- Public key:
  - An encryption of 0: n elements that generate G
  - An encryption of 1: n elements that generate H
- Secret key: Trapdoor to distinguish G from H (represented by generators).
- Encryption: Randomize the encryption of 0 or 1.

- AND gate: Given generators of groups K1, K2, output generators of the union of K1,K2. (Use union of generators.)

- OR gate: Given generators of groups K1,K2, output generators of intersection of K1,K2. (Use commutator.)
  - G = [G,G], H = [H,H], H = [G,H].

Existence?

- Need perfect group pairs with hard distinguishing problem (and efficient operations and a trapdoor)

- Example of perfect group pair with easy dist. problem:
  - Direct product: $G = H \times K$, where H and K are perfect

Failed Attempt

Form of G elements

$$R \cdot \begin{bmatrix} * & * & 0 & 0 & 0 & 0 & 0 & 0 \\ * & * & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & * & * & * & * & * & * \\ 0 & 0 & * & * & * & * & * & * \\ 0 & 0 & * & * & * & * & * & * \\ 0 & 0 & * & * & * & * & * & * \\ 0 & 0 & * & * & * & * & * & * \\ 0 & 0 & * & * & * & * & * & * \end{bmatrix} R^{-1}$$

Form of H elements

$$R \cdot \begin{bmatrix} * & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & * & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & * & * & * & * & * & * \\ 0 & 0 & * & * & * & * & * & * \\ 0 & 0 & * & * & * & * & * & * \\ 0 & 0 & * & * & * & * & * & * \\ 0 & 0 & * & * & * & * & * & * \\ 0 & 0 & * & * & * & * & * & * \end{bmatrix} R^{-1}$$

- Linear algebra attack: Encryptions of 0 in proper subspace
- Is there a patch? Can we use non-abelian *groups* without fatally embedding them in a *ring*? (representation theory)

Thank You! Questions?

Barrington and Non-Abelian Groups

- NC1 circuits to a product of permutations
- On each circuit wire w:
  - "0" is represented by the identity permutation ε
  - "1" is represented by some non-identity permutation π_w
- $\text{AND}(w1,w2) = \pi_{w1} \circ \pi_{w2} \circ \pi_{w1}^{-1} \pi_{w2}^{-1}$
  - Equals ε ("0") if either w1 or w2 is ε ("0")
  - Equals a non-identity permutation if the inputs are non-commuting non-identity permutations π_{w1} and π_{w2}.

The Noise Problem Revisited

- Ciphertext noise grows exponentially with depth d.
  - Hence log q and dimension of ciphertext matrices grow linearly with d.
- Want overhead to be independent of d.
  - To only depend on the security parameter λ.
- Achievable!
  - Via a technique called bootstrapping [Gentry '09].