Compressible FHE with Applications to PIR

Craig Gentry, Shai Halevi

Algorand Foundation Research

*Work done while at IBM Research

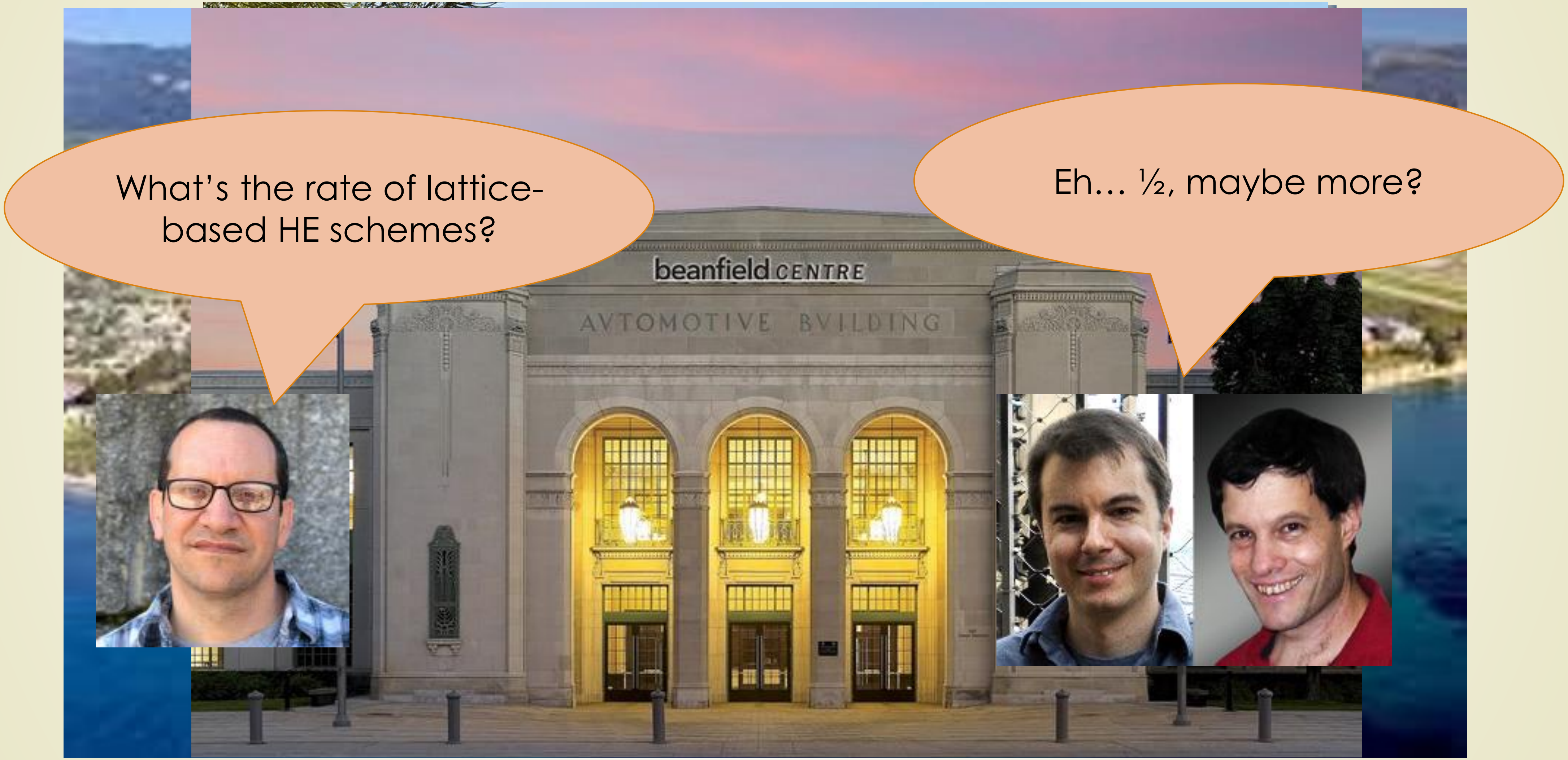
Information Rate of Homomorphic Encryption

- Contemporary (F)HE is a bandwidth hog
  - Ciphertext is larger than plaintext by at least a large constant factor (sometimes more)
- This is NOT the case for standard encryption
  - Can do $|ctxt| \sim |ptxt|$
- Can we hope to get similar efficiency with (F)HE?

Information Rate of Homomorphic Encryption

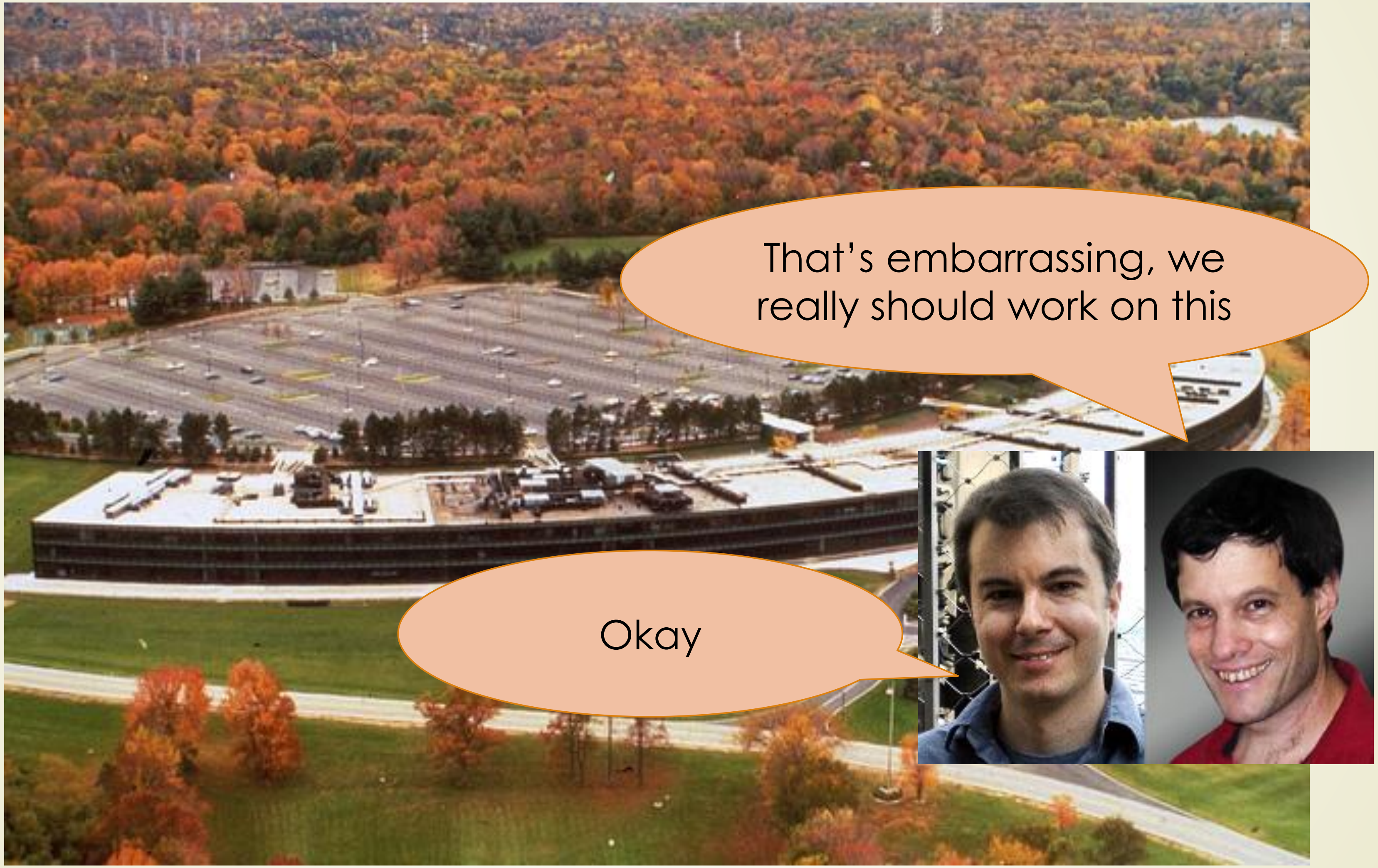
- The only rate-efficient HE is Damgård–Jurik
  - $ptxt \in Z_{N^r},\ ctxt \in Z_{N^{r+1}}$, for any desirable r
  - Can grow r to get rate 1-ε for any ε>0
- But
  - only additive-homomorphic
  - rather slow (especially in the context of applications)
  - not quantum safe
- What about lattice-based HE schemes?

History of This Work



York Berkeley, 2018 2017

History of This Work



Back in Yorktown Heights, 2018

This Work

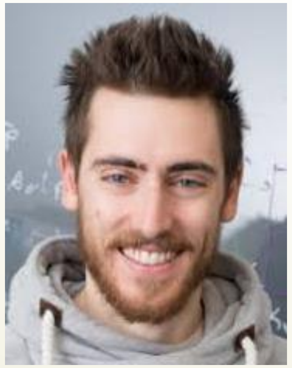
- A “compressible” LWE-based (F)HE
  - Rate 1-ε, security under LWE with gap $\lambda^{O(1/\epsilon)}$
- Application to single-server PIR
  - First “practical” scheme for large databases
  - Rate 4/9, should be 10-20 cycles per byte in db
    - Faster than whole-database AES encryption
  - Compare to state of the art (SealPIR, [ACLS18]), with rate 1/1000 and >100 cycles/byte

Meanwhile, elsewhere…

I’d better ask someone else







Independent Work

- Döttling, Garg, Ishai, Malavolta, Mour, Ostrovsky. *Trapdoor hash functions and their applications*. CYRPTO 2019.
  - Limited homomorphism, choice of assumptions
- Brakerski, Döttling, Garg, Malavolta. *Leveraging linear decryption: Rate-1 fully-homomorphic encryption and time-lock puzzles*. 2019.
  - FHE, based on LWE
- More general than ours, less practically efficient

What is Compressible (F)HE?

- An (F)HE Scheme **(KeyGen, Enc, Dec, Eval)**
  - Encrypted output $\leftarrow$ Eval(circuit, encrypted Input)
- But Dec is broken into:
  - **Compression**: c* $\leftarrow$ Compress(c1,c2,…)
  - **Compressed decryption**: m1,m2,…$\leftarrow$ cDec(c*)

Rate α: For any circuit Π with long enough output

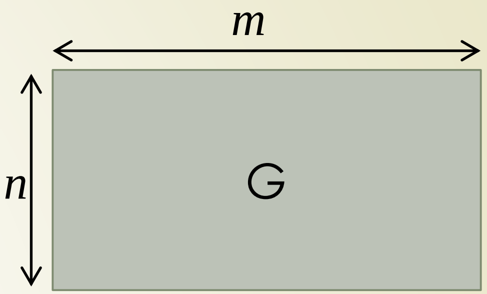
|Compress(Eval(Π, Enc(input)))|<| Π output|/α

Background: [PVW08] Packing

- Recall Regev encryption
  - A $(\lambda + 1)$ (pseudorandom) vector encrypts one scalar
  - $\boldsymbol{\langle(\overrightarrow{sk}| - 1), \overrightarrow{ct}\rangle = encode(m) + e \ (mod\ q), \ \ |e| \ll q}$
- [PVW08]: Regev-like with rate 1-ε
  - A $(\lambda + r)$ (pseudorandom) vector encrypts r scalars
    - Can grow r to get rate 1-ε for any ε>0
  - $\boldsymbol{[S| - I] \cdot \overrightarrow{ct} = encode(\overrightarrow{m}) + \vec{e} \ (mod\ q), \ \ |\vec{e}| \ll q}$
    - Each row of this equation is a Regev encryption

Background: "Gadget Matrices" [MP12]

- A rectangular matrix $G \in Z_q^{n\times m}$
  - A known "public trapdoor" $G^{-1}(0) \in Z_q^{m\times m}$:
    a. Entries of $G^{-1}(0)$ are small, $|G^{-1}(0)|_\infty \ll q$
    b. $G^{-1}(0)$ has full rank over the reals
    c. $G \times G^{-1}(0) = 0\ (mod\ q)$



- For $C \in Z_q^{n\times m}$, $G^{-1}(C)$ is a redundant version of C
  - An $m \times m$ matrix satisfying a,b, and $\mathrm{G} \times G^{-1}(C) = C$
  - Can be found efficiently from C
  - The more rectangular G, the smaller $|G^{-1}(\cdot)|$ can get

Background: “Gadget Matrices” [MP12]

- Example, bit-decomposition: $\ell = \lfloor \log q \rfloor,\ m = n \cdot \ell$

$$G = \begin{bmatrix} 1\ 2 \ldots 2^{\ell-1} & & & \\ & 1\ 2 \ldots 2^{\ell-1} & & \\ & & \ddots & \\ & & & 1\ 2 \ldots 2^{\ell-1} \end{bmatrix}$$

$$G^{-1}(0) = \left[\begin{array}{ccccc|c} 2 & & & & & \\ -1 & 2 & & & & \\ & -1 & & & & \text{bits-of-}q \ldots \text{bits-of-}q \\ & & \ddots & 2 & & \\ & & & -1 & & \end{array}\right]$$

Background: [GSW13] HE Scheme

- Ptxt: scalars (e.g., bits), Ctxt: $n \times m$ matrices
- $C \in Z_q^{n \times m}$ encrypts $\sigma \in Z_q$ wrt $\overrightarrow{sk}$ if

$$\overrightarrow{\boldsymbol{sk}} \cdot \boldsymbol{C} = \boldsymbol{\sigma} \cdot \overrightarrow{\boldsymbol{sk}} \cdot \boldsymbol{G} + \vec{\boldsymbol{e}}\ (\boldsymbol{mod}\ \boldsymbol{q}) \quad |\vec{\boldsymbol{e}}| \ll \boldsymbol{q}$$

- $C_1 + C_2$ encrypts $\sigma_1 + \sigma_2$
- $C_1 \cdot G^{-1}(C_2)$ encrypts $\sigma_1 \sigma_2$
  - Multiplication noise term is $\sigma_1 \cdot \vec{e}_2 + \vec{e}_1 \cdot G^{-1}(C_2)$
  - The scalars σ should be small

Our Construction

The Two Parts of Our Compressible HE

- Low-rate scheme for homomorphism
  - A slight variant of GSW
- High-rate scheme for compression
  - Somewhat similar to the matrix HE scheme of [HAO16]
  - Ptxt, ctxt are matrices of similar dimensions
  - We describe two variants of that scheme
- The two parts “play nice” together
  - They share the same secret key
  - Can pack many GSW ctxts in one high-rate ctxt

The Low-Rate Scheme

- Like GSW, but sk is a matrix, $S = [S' | -I]$
  - As in [PVW08]
- If $C \in Z_q^{n\times m}$ encrypts $\sigma \in Z_q$ then

$$\boldsymbol{S \cdot C = \sigma \cdot S \cdot G + \vec{E}\ (mod\ q) \quad |\vec{E}| \ll q}$$

  - Each row is a GSW invariant, all with the same σ
- Homomorphic operations work exactly as in GSW
  - $C_1 + C_2$ encrypts $\sigma_1 + \sigma_2$, $C_1 \cdot G^{-1}(C_2)$ encrypts $\sigma_1\sigma_2$
    - Multiplication noise term is $\sigma_1 \cdot \vec{E}_2 + \vec{E}_1 \cdot G^{-1}(C_2)$

The High-Rate Scheme

- Ctxt C encrypts ptxt M wrt S if

$$\boldsymbol{S \cdot C = \mathbf{encode}(M) + E \; (mod\; q)} \quad \boldsymbol{|\mathrm{E}| \ll q}$$

  - Encoding is needed to remove noise E on decryption
- Two variants, differ in how they encode M
- One uses a “nearly square” new gadget matrix
  - Ptxt, ctxt are both matrices modulo q
- Another variant uses scaling instead
  - Ptxt are matrices modulo some $p < q$

A Nearly-Square Gadget Matrix

- To get high rate, we want to add "just a little redundancy", enough to remove a little noise
  - Want "only a little rectangular" gadget matrix H
- Consider what we need from $F = H^{-1}(0)$:
  - It needs to be at least somewhat small
  - It should have full rank over the reals
  - But also $\mathrm{H} \times F = 0 \ (mod\ q)$
    - So F only has a very small rank modulo q
    - Recall that H is nearly-square

A Nearly-Square Gadget Matrix

- Example when $q = p^t - 1$ for some integers p, t
  - Let $F = \begin{bmatrix} 1 & p & p^2 & & p^{t-1} \\ p^{t-1} & 1 & p & & p^{t-2} \\ p^{t-2} & p^{t-1} & 1 & & p^{t-3} \\ & & & \ddots & \\ p & p^2 & p^3 & & 1 \end{bmatrix}$
  - $|\mathrm{F}|$ is small enough to remove noise of size upto $\frac{p-1}{2}$
  - F has full rank over the reals, only rank one mod q
- $H \in Z_q^{(t-1)\times t}$ is any basis of the null space of F mod q
  - Can use $H_r = H \otimes I_r$ (for any r), with $F_r = H_r^{-1}(0) = F \otimes I_r$

Can relax $q = p^t - 1$ to $q = p^t - \alpha$ for small α

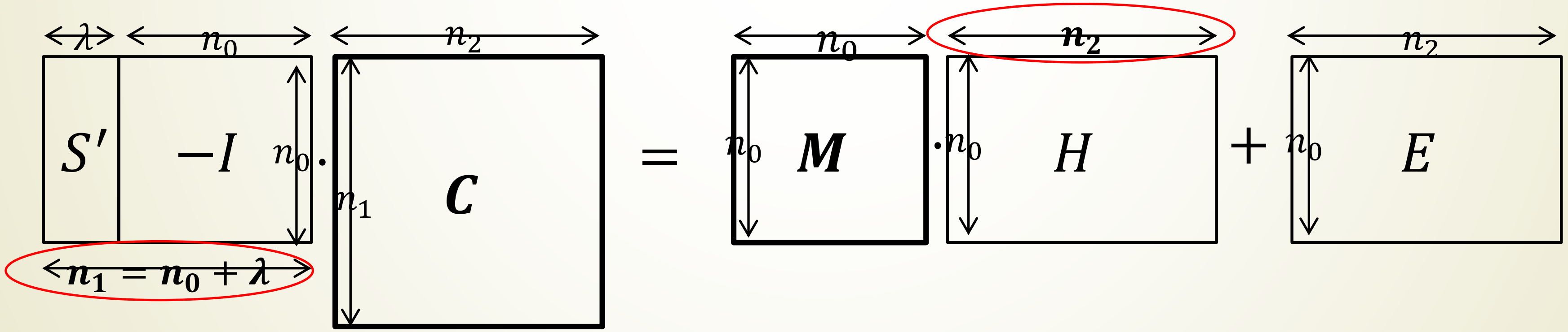
The High-Rate Scheme (1st Variant)

- Ctxt $C \in Z_q^{n_1 \times n_2}$ encrypts ptxt $M \in Z_q^{n_0 \times n_0}$ wrt S if

$$\boldsymbol{S \cdot C = M \cdot H + E \ (mod\ q)} \quad |\mathbf{E}| \ll \boldsymbol{q}$$

  - E is small enough so H can be used to remove it

- Note the dimensions of the various matrices



  - Rate is $n_0^2 / n_1 n_2$

The High-Rate Scheme (1st Variant)

- Ctxt $C \in Z_q^{n_1 \times n_2}$ encrypts ptxt $M \in Z_q^{n_0 \times n_0}$ wrt S if

$$\boldsymbol{S \cdot C = M \cdot H + E \ (mod\ q) \quad |E| \ll q}$$

  - E is small enough so H can be used to remove it

- Compressed Decryption:
  - $X := S \cdot C = M \cdot H + E \ (mod\ q)$
  - $Y := X \cdot F = E \cdot F \ (mod\ q)$
    - Since $H \cdot F = 0 \ (mod\ q)$
  - If $|E \cdot F| < q/2$ then $Y = E \cdot F$ over the integers
  - Can multiply by F^{-1} to recover E, then remove it

Compression

- Consider many GSW bit encryptions

$$\boldsymbol{S \cdot C_{u,v,w} = \sigma_{u,v,w} \cdot S \cdot G + E_{u,v,w}}$$

  - $u, v \le n_0,\ w \le \ell = \log q$
  - Enough bits $\sigma_{u,v,w}$ for a plaintext matrix $M \in Z_q^{n_0 \times n_0}$
- Let $T_{u,v}$ *be the* $n_0 \times n_0$ *singleton matrix* $e_u \otimes e_v$
  - *1 only in entry* u, v, 0 elsewhere
  - Also let $T'_{u,v} = \begin{array}{cc} n_0 & \\ \begin{bmatrix} 0 \\ \hline -T_{u,v} \end{bmatrix} & \begin{array}{c} \lambda \\ n_0 \end{array} \end{array} \in Z_q^{n_1 \times n_0}$

$\boxed{\text{Note } [S' | {-I}] \cdot T'_{u,v} = T_{u,v}}$

Compression

- To pack all the GSW ciphertexts $C_{u,v,w}$ we set

$$\boldsymbol{C^* = \sum_{u,v,w} \underbrace{C_{u,v,w}}_{n_1 \times m} \cdot \underbrace{G^{-1}\left(2^w \cdot T'_{u,v} \cdot H\right)}_{m \times n_2} \ (mod\ q)}$$

- $S \cdot C^* = \sum S \cdot C_{u,v,w} \cdot G^{-1}\left(2^w \cdot T'_{u,v} \cdot H\right)$

$$= \sum (\sigma_{u,v,w} \cdot S \cdot G + E_{u,v,w}) \cdot G^{-1}\left(2^w \cdot T'_{u,v} \cdot H\right)$$

$$= \sum 2^w \cdot \sigma_{u,v,w} \cdot S \cdot T'_{u,v} \cdot H \ + \ noise$$

$$= \underbrace{\left(\sum_{u,v} \underbrace{\left(\sum_w 2^w \cdot \sigma_{u,v,w}\right)}_{z_{u,v}} \cdot T_{u,v}\right)}_{M} \cdot H + noise$$

The High-Rate Scheme (2nd Variant)

- Ctxt $C \in Z_q^{n_1 \times n_0}$ encrypts ptxt $M \in Z_p^{n_0 \times n_0}$ wrt S if

$$\boldsymbol{S \cdot C = \lfloor ^q/_p \rceil \cdot M + E \ (mod\ q)} \quad |\mathbf{E}| < {}^{\boldsymbol{q}}/_{\boldsymbol{2p}}$$

  - $p < q$, but close (say $p = q^{1-\epsilon}$)
- Use scaling to remove noise on decryption
- Compression is similar to before
  - Except that $G^{-1}\left(2^w \cdot T'_{u,v} \cdot H\right)$ is replaced by $G^{-1}\left(2^w \cdot \lfloor ^q/_p \rceil \cdot T'_{u,v}\right)$.

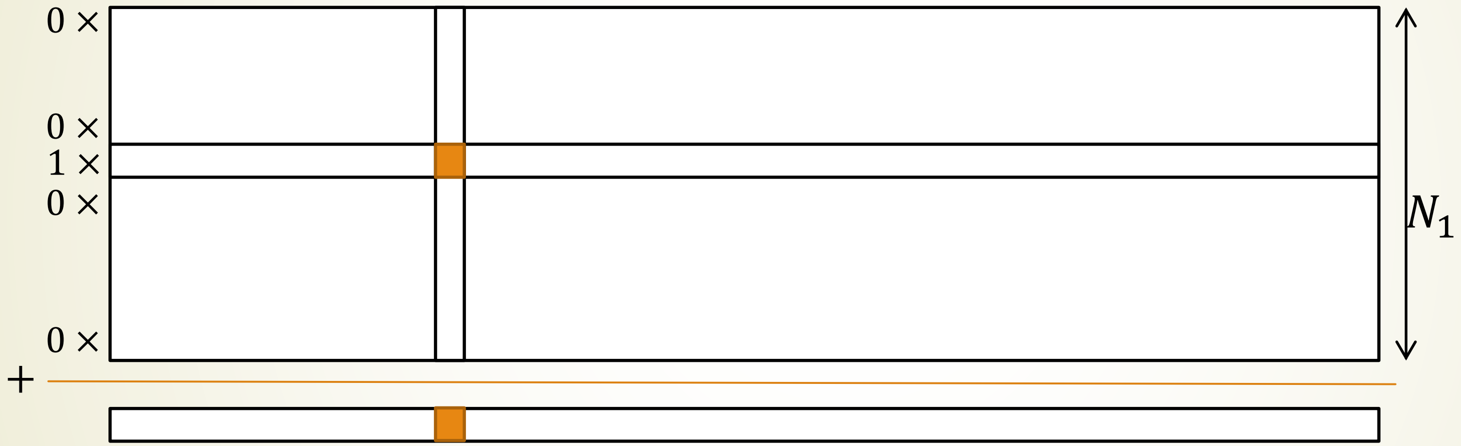
Single Server PIR

Application to Single-Server PIR

- Compressible HE easily yields high-rate PIR
- But we also want practical efficiency

Our Approach to Single-Server PIR

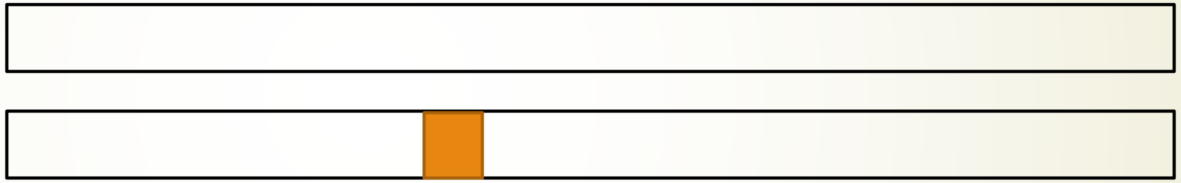
- Start from the basic scheme of [KO97]
  - Think of N-entry DB as an $N_1 \times {}^{N}/_{N_1}$ matrix



  - Continue recursively on the ${}^{N}/_{N_1}$-database
  - Almost all the work is in the 1st step

A Few More Pieces of Magic

- Multiplying a GSW ctxt by high-rate ctxt yields a high-rate ciphertext of the product
  - Same for multiplying a GSW ctxt by plaintext M
- The products $0 \times$ / $1 \times$



  yield high-rate encryption of the database
  - High-rate scheme is additively homomorphic
  - All we need is to add across the 1st dimension
- The same holds for the recursive levels

From Here to Practical Single-Server PIR

- Many more tricks
- Pre-processing the db to eliminate FFTs
- Switching to RLWE
- Different gadget matrices G in different steps
- Using modulus switching
- …

The End-Result PIR

- Rate is $(2/3)^2 = 4/9$
  - S is a 2-by-3 matrix (over a ring)
  - H is a 2-by-3 matrix (over a ring)

$$\boxed{S \cdot \overbrace{C}^{3\text{x}3} = \overbrace{M}^{2\text{x}2} \cdot H + E}$$

- Total work ~ 1.5 multiplies per database byte
  - Modulo single-precision numbers (upto 60 bits)
  - Should be 10-20 cycles per byte in software

The End-Result PIR

- First single-server PIR plausibly efficient enough to handle large databases
- Less work than whole database AES encryption
  - Which you would need (for communication security) if you used the naïve solution
  - So we beat the naïve solution not only on bandwidth but also on server computation


"That's all Folks!"