Client-Optimized Algorithms & Acceleration for Encrypted Compute Offloading

McKenzie van der Hagen – mckenziv@andrew.cmu.edu
Brandon Lucia – blucia@andrew.cmu.edu
Client-aided HE for Opaque Compute Offloading

- CHOCO enables privacy-preserving computation for resource-constrained devices
- CHOCO utilizes Homomorphic Encryption (HE) and Client-Aided Encrypted Computing
- CHOCO introduces client-optimized encrypted algorithms & hardware acceleration
- CHOCO makes client responsibility competitive with local compute
- CHOCO benefits generalize to diverse applications
Outline

- Introduction
  - Resource-Constrained Devices & Privacy-Preserving Offload

- Encrypted Computing Background
  - Fully Homomorphic Encryption & Client-Aided Encrypted Computing

- CHOCO System Implementation
  - Algorithm Optimizations & Hardware Acceleration

- Results & Conclusions
  - Active Client Computation, Communication & Applications
Outline

• Introduction
  • Resource-Constrained Devices & Privacy-Preserving Offload

• Encrypted Computing Background

• CHOCO System Implementation

• Results & Conclusions
Resource-Constrained Devices are Everywhere

Health Monitoring

Wildlife Monitoring

Infrastructure Monitoring
Computational Demands Exhaust Sensor Resources

- Device Size
- Data Quantity
- Computation Complexity

- 114 mg/dL
- 109 / 72
- 97% SpO2
- 78 BPM
Privacy-Preserving Computation Offload

**Communication**
Transfer Data & Results

**Shared Offload Server**
Semi-Honest Central Compute

**Resource-Constrained Private Client**
Secure Data Collection

78 BPM
Privacy-Preserving Computation Offload

Communication
Transfer Data & Results

Shared Offload Server
Semi-Honest Central Compute

Resource-Constrained Private Client
Secure Data Collection

FHE: [Fully] Homomorphic Encryption
Outline

• Introduction
  • Resource-Constrained Devices & Privacy-Preserving Offload

• Encrypted Computing Background

• CHOCO System Implementation

• Results & Conclusions
Outline

• Introduction

• Encrypted Computing Background
  • Fully Homomorphic Encryption & Client-Aided Encrypted Computing

• CHOCO System Implementation

• Results & Conclusions
Homomorphic Encryption (HE)

Plaintext Input Vectors

15, 6, 20  3, 14, 0

Encrypt

100011001  ×  011101010

= 100011001

Decrypt

976, 43, 2342

Incorrect output due to noise growth
Polynomial Operations

\[ ct_a \times ct_b = (c_0^a, c_1^a) \times (c_0^b, c_1^b) \]

\[ = \left( [c_0^a \times c_0^b]_q , [c_0^a \times c_1^b + c_1^a \times c_0^b]_q , [c_1^a \times c_1^b]_q \right) \]

Polynomial Multiplication & Addition with Thousands of Coefficients

Firsts

Outers

Inners

Lasts

= (c_0^c, c_1^c) \rightarrow \text{Relinearization} \rightarrow (c_0^c, c_1^c) = ct_c = \left( \begin{array}{c}
100011001
\end{array} \right)
HE Challenges & Limitations

- High Computation Costs
- Linear Operations
- Noise Growth & Arithmetic Depth
- Parameter Selection
HE Challenges & Limitations

- High Computation Costs
- Linear Operations
- Noise Growth
- Parameter Selection

Offloading Entire Encrypted Applications is Still Infeasible in Many Scenarios
Client-Aided Encrypted Computing
Client-Aided Encrypted Inference

- Systematically limits arithmetic depth & regularly refreshes noise

Quantifying Client Responsibility

- ARM Cortex-A7 CPU Client
- **Up to 180x** overhead to offload compute
- Dominated by Homomorphic Encryption (HE) operations

Microsoft Research. 2019. *Simple Encrypted Arithmetic Library (release 3.4)*. [https://github.com/Microsoft/SEAL](https://github.com/Microsoft/SEAL)
Quantifying Client Responsibility

- ARM Cortex-A7 CPU Client
- **Up to 180x** overhead to offload compute

Dominated by Homomorphic Encryption (HE) operations

**CHOCO** Reduces Client-Side Computation by up to 341x through SW Algorithms & HW Acceleration
Complete Client-Aided System Improvements

HE Inference Execution Time for SqueezeNet-CIFAR

<table>
<thead>
<tr>
<th>Component</th>
<th>CHOCO</th>
<th>Server-Optimized Solutions [Samardzic '21]</th>
<th>TFLite</th>
</tr>
</thead>
<tbody>
<tr>
<td>Communication</td>
<td>5.01</td>
<td>14.50</td>
<td>0.01</td>
</tr>
<tr>
<td>Server Computation</td>
<td>0.01</td>
<td>75.02</td>
<td>0.06</td>
</tr>
<tr>
<td>Client Computation</td>
<td>21.81</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Outline

- Introduction
- Encrypted Computing Background
  - Fully Homomorphic Encryption & Client-Aided Encrypted Computing
- CHOCO System Implementation
- Results & Conclusions
Outline

• Introduction

• Encrypted Computing Background

• CHOCO System Implementation
  • Algorithm Optimizations & Hardware Acceleration

• Results & Conclusions
Client-aided HE for Opaque Compute Offloading

Communication
Encrypted Intermediate Results

Shared Offload Server
Encrypted Linear Operations

Private Client
Plaintext Non-linear Operations
Client-aided HE for Opaque Compute Offloading

Hardware Acceleration
Encryption & Decryption

Communication
Encrypted Intermediate Results

Shared Offload Server
Encrypted Linear Operations

Private Client
Plaintext Non-linear Operations

HE Algorithm Optimization
Minimize Arithmetic Depth
Encrypted Algorithm Optimization
HE Algorithms

1. Input Packing

2. HE Operation Scheduling

3. Output Unpacking

Client

Offload Device
Algorithm Optimizations Impact Client Costs

- Minimize Arithmetic Depth
- Noise Growth
- Parameter Selections
- Ciphertext Size
- Computation & Mem
- Communication
Windowed Rotations

```
1 2 3 4
```

Rotate << 2

```
3 4 1 2
```

```
+ + + +
```

```
28 74 62 9
```

Windowed Rotations

Ideally: Values wrap around within a window of interest
Windowed Rotations

Actually: Values wrap around the entire ciphertext vector
Standard Permutations

- Expensive
- Computation & Noise Growth
- High Arithmetic Depth
Rotational Redundancy

- Novel Input Packing
- Single HE Rotation
- Low arithmetic depth
CHOCO Algorithms Reduce Client Computation

**CHOCO SW** = SEAL baseline + Rotational Redundancy

- 50% Smaller Ciphertexts
- Average **1.7x** improvement over SEAL
- **1.87x** improvement over SqueezeNet-CIFAR
- **96.3x** remaining overhead vs TFLite

- **Average **62.5x** remaining overhead vs TFLite**
CHOCO Algorithms Reduce Client Computation

CHOCO Algorithm Optimizations Provide a Critical but Insufficient Reduction in Client Computation

- **CHOCO SW** = SEAL baseline + Rotational Redundancy
- Average 1.7x improvement over SEAL
- 50% Smaller Ciphertexts
- Average 62.5x remaining overhead vs TFLite
Hardware Acceleration
Existing FPGA Acceleration is Incomplete

• FPGA HW = CHOCO SW + Encryption/Decryption FPGA

• Average 14.5x remaining overhead vs TFLite

CHOCO – Through Accelerated Cryptographic Operations
• Encrypt(pk, m): For $m \in R_d$, let $pk = (p_0, p_1)$. Sample $u \leftarrow R_2$, and $e_1, e_2 \leftarrow \chi$. Compute

$$ct = ([\Delta m + p_0 u + e_1]_q, [p_1 u + e_2]_q).$$
CHOCO-TACO Encryption & Decryption Hardware

- Encrypt(pk, m): For $m \in R_d$, let $pk = (p_0, p_1)$. Sample $u \leftarrow R_2$, and $e_1, e_2 \leftarrow \chi$. Compute

$$ct = (\lfloor \Delta m + p_0 u + e_1 \rfloor_q, \lfloor p_1 u + e_2 \rfloor_q).$$
CHOCO-TACO Encryption & Decryption Hardware
CHOCO-TACO Hardware Optimization
CHOCO-TACO Hardware Optimization

Poly Multiplication Module

**Modules & Functional Blocks**  
Specialized & Comprehensive
CHOCO-TACO Hardware Optimization

- Poly Multiplication Module
  - NTT Block
  - Pairwise Product Block
  - INTT Block

Modules & Functional Blocks
Pipelined & Comprehensive
CHOCO-TACO Hardware Optimization

Conceptual Layers
Polynomial Parallelism

Poly Multiplication Module

- NTT Block
- Pairwise Product Block
- INTT Block

Modules & Functional Blocks
Specialized & Comprehensive
CHOCO-TACO Hardware Optimization

Poly Multiplication Module

- NTT Block
- Pairwise Product Block
- INTT Block

Modules & Functional Blocks
Specialized & Comprehensive

Conceptual Layers
Polynomial Parallelism

Processing Elements
Coefficient Parallelism
CHOCO-TACO Hardware Optimization

**Poly Multiplication Module**
- **NTT Block**
  - Twiddle Factors
- **Pairwise Product Block**
- **INTT Block**
  - Twiddle Factors

**Modules & Functional Blocks**
- Specialized & Comprehensive

**SRAM Buffers**
- Parallelism, Pipelining, Minimal Data Movement

**Conceptual Layers**
- Polynomial Parallelism

**Processing Elements**
- Coefficient Parallelism
CHOCO-TACO Hardware Optimization

Full Design Space Study in Paper!
CHOCO-TACO Encryption & Decryption Hardware

19.3 mm² area. Consumes 200 mW power, .1228 mJ to perform a single encryption in .66 ms.
Outline

- Introduction
- Encrypted Computing Background
- CHOCO System Implementation
- Results & Conclusions
  - Active Client Computation, Communication & Applications
CHOCO-TACO Accelerates Client Compute

CHOCO HW = CHOCO SW + CHOCO-TACO Encryption/Decryption Simulated ASIC

Average 123.3x Improvement over CHOCO software alone
CHOCO-TACO Accelerates Client Compute

Active Client Computation
- Encryption
- Decryption
- ML Computation & Quantization

- **LeNet-5-Small**
  - With CHOCO HW: 0.133
  - With TFLite Baseline: 0.004
  - With FPGA HW: 0.007

- **LeNet-5-Large**
  - With CHOCO HW: 0.495
  - With TFLite Baseline: 0.031
  - With FPGA HW: 0.047

- **SqueezeNet-CIFAR**
  - With CHOCO HW: 2.660
  - With TFLite Baseline: 0.064
  - With FPGA HW: 0.121

- **VGG16-CIFAR**
  - With CHOCO HW: 4.545
  - With TFLite Baseline: 0.193
  - With FPGA HW: 0.717

- Average **29x** better than FPGA accelerators
- Average **2.2x** better than local compute via
CHOCO Makes Client End-to-End Costs Feasible

Local Compute w/ TFLite vs Offloaded Compute w/ CHOCO
CHOCO Makes End-to-End Client Costs Feasible

- Privacy-Preserving Offload can be **Competitive with Local Compute**
- **37%** decrease in energy consumption for VGG16
- Up to **66% communication reduction** from SEAL baseline
CHOCO Algorithms Reduce Communication

- Up to three orders of magnitude improvement in communication
- Nearly **90x** improvement over Gazelle [Juvekar `18]
- **28x** better than LoLa (*not* client-aided) [Brutzkus `19]

---

See Paper for More Applications & Results

Unmodified DNN Networks

Encrypted Distance Calculations
(K-Means & KNN)
Conclusions

- CHOCO motivates and prioritizes **client-aware optimizations**
- CHOCO algorithm optimizations **reduce communication by orders of magnitude** over prior work
- CHOCO-TACO hardware **comprehensively accelerates client-side cryptographic primitives**
- CHOCO **enables participation from resource-constrained devices** in client-aided encrypted computation
- CHOCO makes client responsibility **competitive with local compute**
- CHOCO benefits **generalize to diverse applications**
Client-Optimized Algorithms & Acceleration for Encrypted Compute Offloading

Thank You! Questions?

McKenzie van der Hagen – mckenziv@andrew.cmu.edu
Brandon Lucia – blucia@andrew.cmu.edu