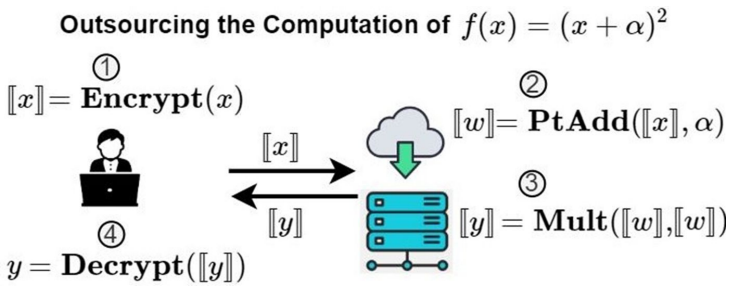
Affordable and Practical Acceleration of CKKS-based Fully Homomorphic Encryption

Rashmi Agrawal, Leo de Castro, Rabia Yazicigil,
Anantha Chandrakasan, Vinod Vaikuntanathan, Chiraag Juvekar, Ajay Joshi
rashmi23@bu.edu

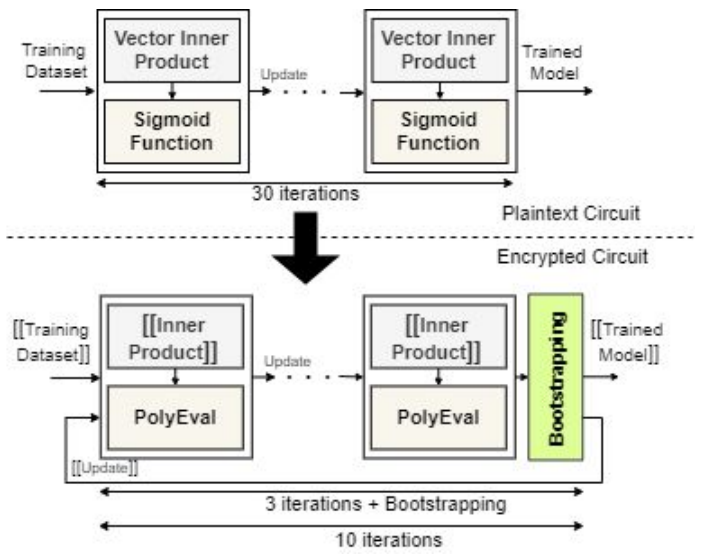
Fully Homomorphic Encryption (FHE)

- FHE enables computation over encrypted data.



FHE Applications

- Medical, Financial, Supply chain, Marketing, etc.
- Machine learning: Logistic regression training



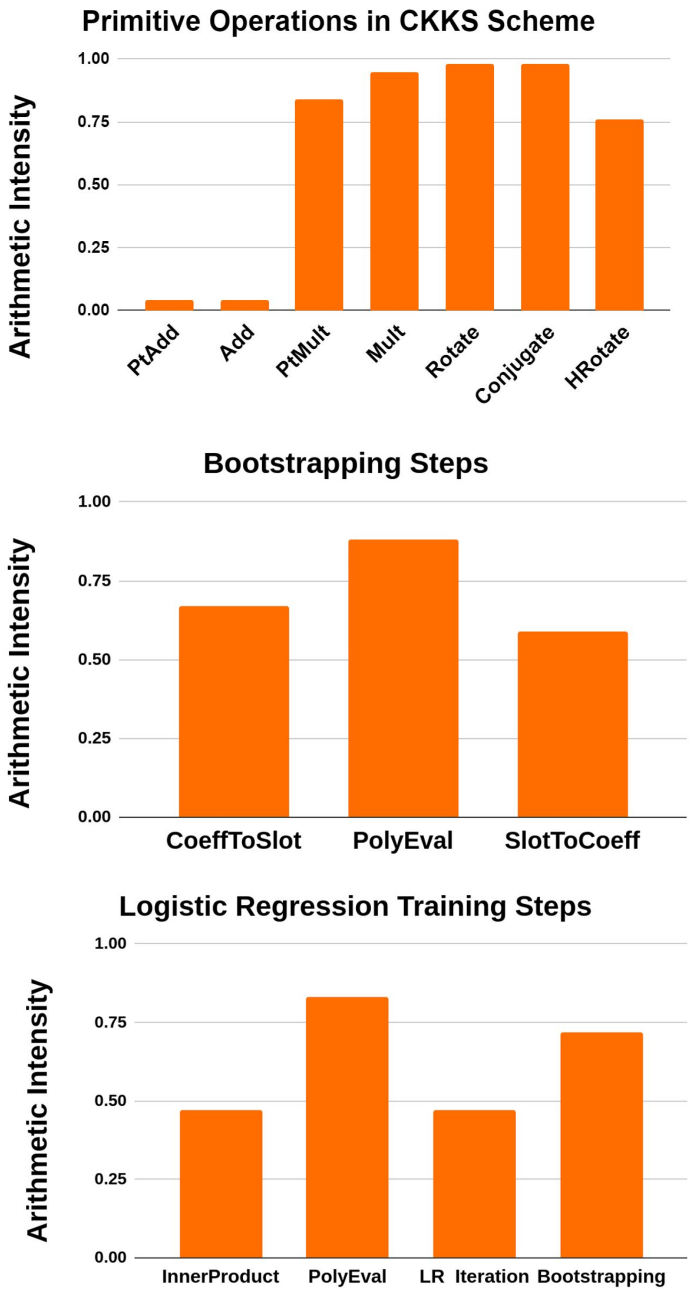
- **Problem:** Bootstrapping is the major bottleneck.

SimFHE: Custom Simulator

- SimFHE:
  - A python-based architectural simulator for modeling CKKS operations at subroutine level.
  - Set scheme parameters.
  - Set architecture parameters.
  - Keeps track of compute operations as well as DRAM transfers for a given cache size.
- Study compute & memory trade-off.
- Explore various optimizations and select parameters to optimize throughput.

Analysis using SimFHE

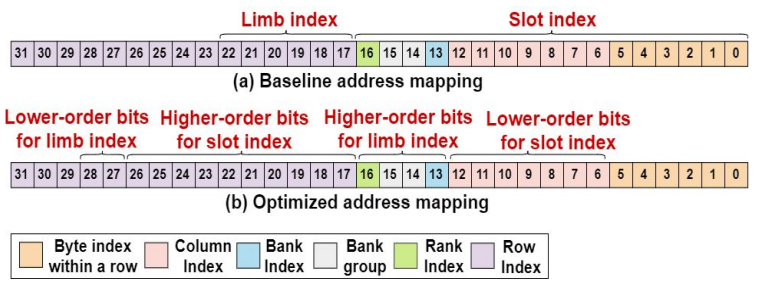
- Arithmetic intensity analysis for CKKS operations and an end-to-end application.
  - < 1 Op/byte



- Bootstrapping has low arithmetic intensity.
- Limited memory bandwidth is the bottleneck.

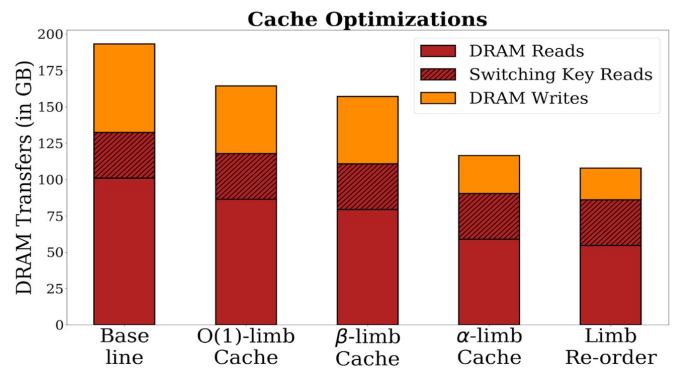
Memory-Aware Design (MAD) Techniques

- Memory access pattern optimization:
  - Regorganize low-level memory access pattern for faster data access.

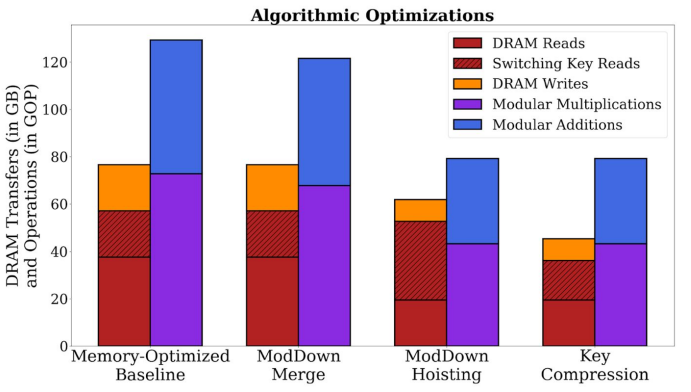


	Limb-wise access	Slot-wise access	Full Switch
Baseline	**2.3 ms**	**9.2 ms**	**11.5 ms**
Optimized	**2.5 ms**	**2.2 ms**	**4.7 ms**

- Hierarchical caching optimizations:
  - Compute as much as possible to maximize data reuse.
  - Re-order operations to maximize cache utilization.

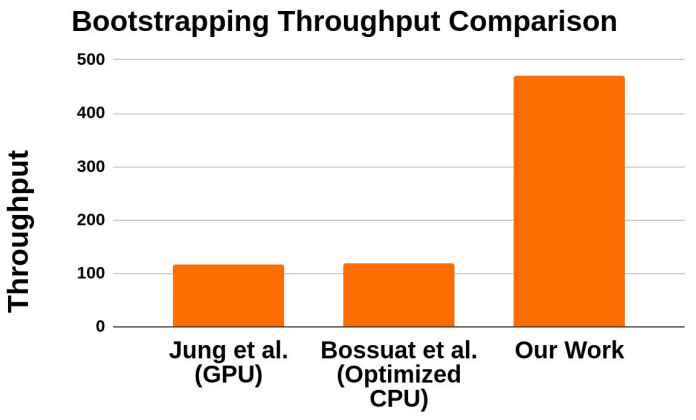


- Algorithmic optimizations:
  - Double-hoisting to reduce orientation switch.
  - Improve arithmetic intensity of low level operations.

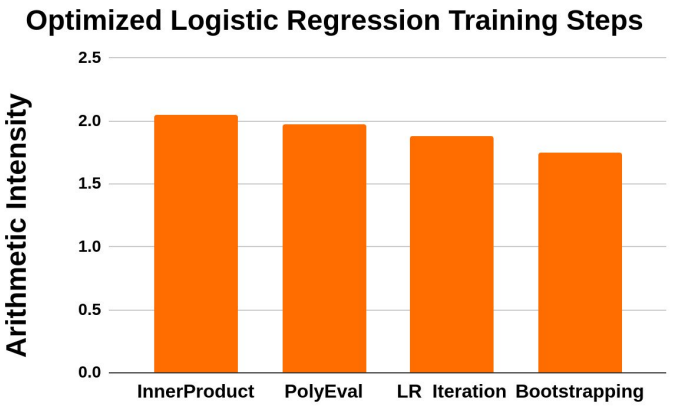


Key Observations

- 8.2x improvement in bootstrapping throughput.



- 3.2x improvement in arithmetic intensity.



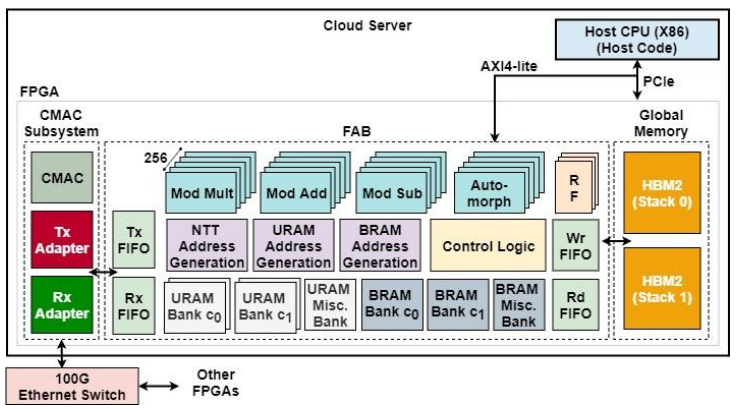
- Memory bandwidth is still a bottleneck.
- CPU/GPU solutions are limited by main memory bandwidth.
- Existing ASIC proposals are too expensive:
  - Need large on-chip memory and register file.
  - Need 12nm/7nm technology nodes.

FAB: FPGA-based Accelerator

- FAB - an FPGA-based accelerator for bootstrappable FHE workloads.
  - First ever bootstrapping implementation on FPGA.
    - Secure and practical parameter set
  - Balanced FPGA design.
    - Not memory-bound.
  - Only 256 functional units operating at 300MHz
  - High data rates from/to main memory at 450MHz.
  - Effective utilization of limited on-chip memory.
    - Modified datapath for basic FHE operations like KeySwitch.
    - Smart Operation scheduling.

FAB: Overall Architecture

- Four components:
  - Host CPU offloads RTL design to the FPGA.
  - RTL design packaged as kernel code.
  - 8GB HBM2 memory stacks.
  - 100G Ethernet CMAC subsystem.



FAB: Evaluation

- Designed in Verilog and implemented on Xilinx Alveo U280 deployed in cloud environment.
- Basic FHE operations' performance:

	FAB	GPU	Speedup (Time)
Add	**0.04 ms**	**0.16 ms**	**3.85x**
Mult	**1.71 ms**	**2.96 ms**	**1.73x**
Rescale	**0.19 ms**	**0.49 ms**	**2.62x**
Rotate	**1.57 ms**	**2.55 ms**	**1.62x**

- Bootstrapping performance:

	Time (micros)	Speedup (Time)	Speedup (Cycles)
CPU	**101.78**	**213x**	**2485x**
GPU	**0.740**	**1.55x**	**6.35x**
FAB	**0.477**	**-**	**-**

- Logistic regression model training with 8 FPGAs:

	Time (sec)	Speedup (Time)	Speedup (Cycles)
CPU	**37.05**	**456x**	**5318x**
GPU	**0.775**	**9.5x**	**39x**
FAB-1	**0.103**	**1.3x**	**1.3x**
FAB-2	**0.081**	**-**	**-**

Conclusion

- Affordable and practical FHE acceleration solution.
  - MAD techniques provide
    - Optimizations feasible with small cache sizes, agnostic of platforms.
  - FAB provides
    - Practical performance using FPGA at a fraction of ASIC cost.