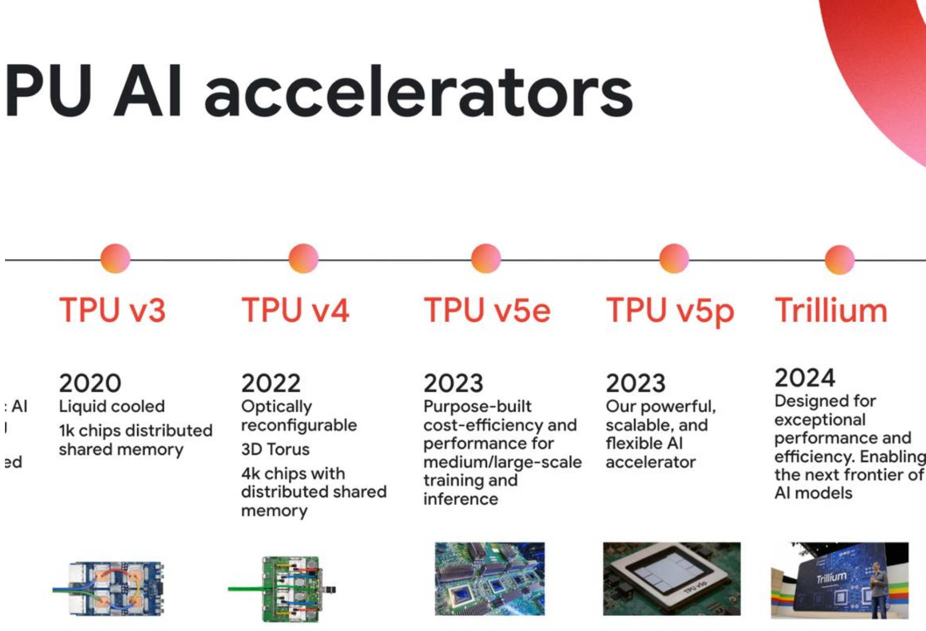
This project is supported by: The NATO Science for Peace and Security Programme

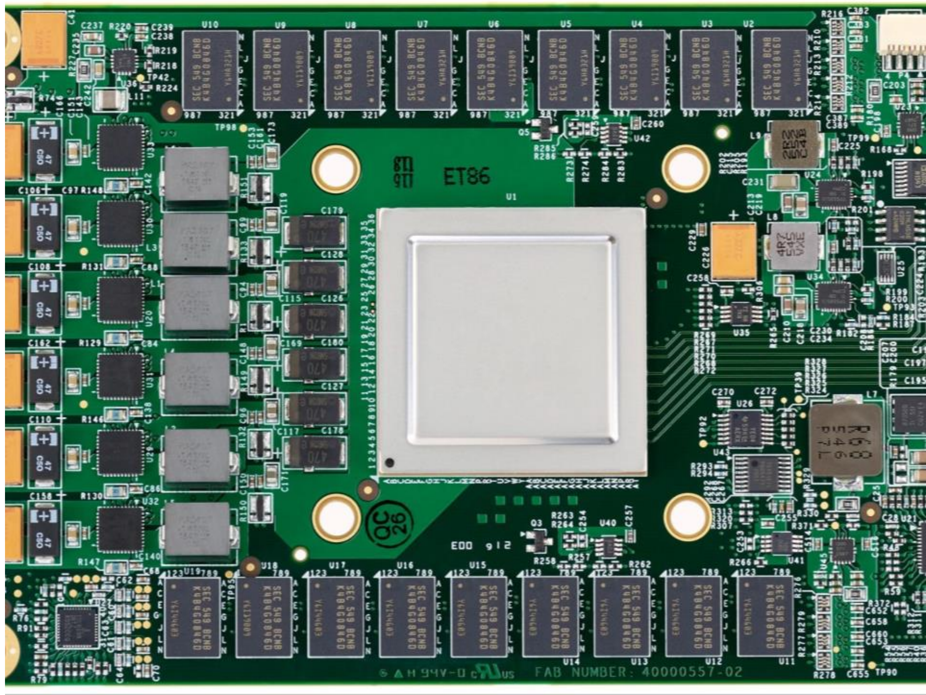
TPU as Cryptographic Accelerator

Rabimba Karanjai, Sangwon Shin, Wujie Xiong, Xinxin Fan, Lin Chen, Tianwei Zhang
Taeweon Suh, Weidong Shi, Veronika Kuchta, Francesco Sica, Lei Xu

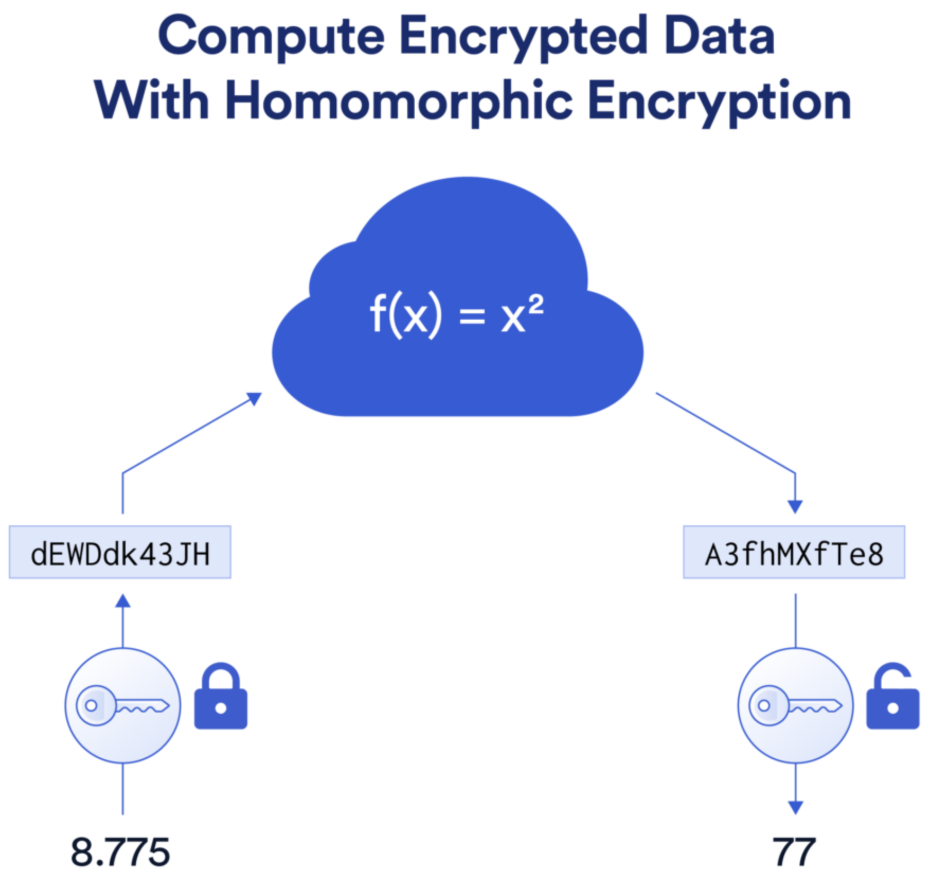
Presented By: Rabimba Karanjai


Trillium
6th generation TPUs


PU AI accelerators
TPU v3
2020
Liquid cooled
1k chips distributed shared memory
TPU v4
2022
Optically reconfigurable
3D Torus
4k chips with distributed shared memory
TPU v5e
2023
Purpose-built cost-efficiency and performance for medium/large-scale training and inference
TPU v5p
2023
Our powerful, scalable, and flexible AI accelerator
Trillium
2024
Designed for exceptional performance and efficiency. Enabling the next frontier of AI models

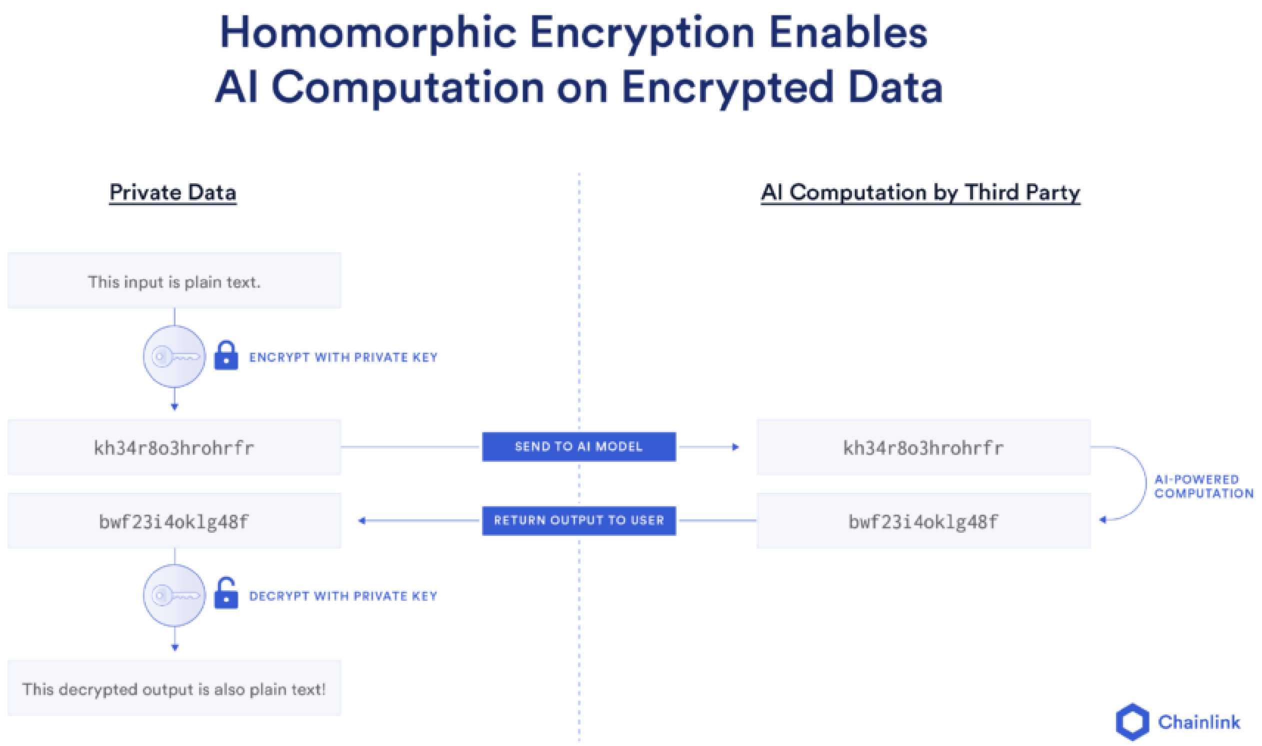


Introduction



Outsource the computation of a function f(x) on data x to a server, without revealing the data to the server.
Source: Chainlink

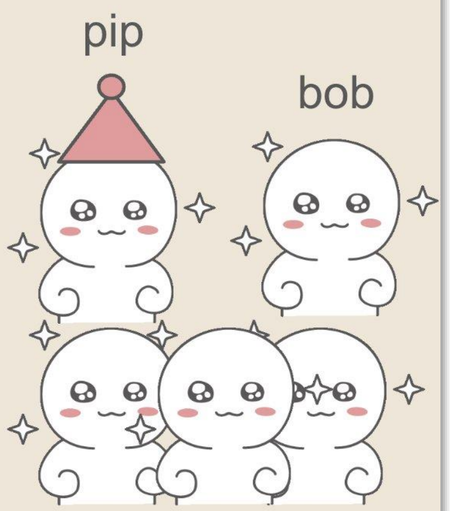
Introduction



FHE

everyone in the group submits their votes but it’s confidential! no one else in the group should know others’ votes or information. everyone just wants to know the voting results.

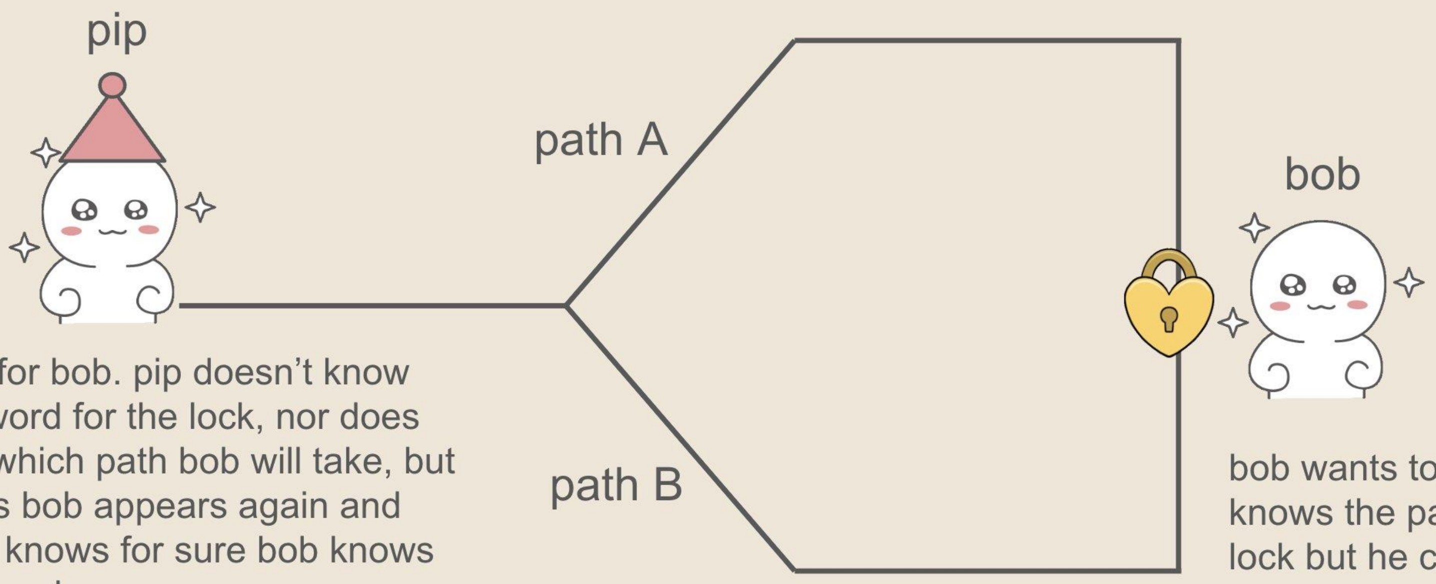
processing votes with information of who voted to produce voting results.



everyone, including pip and bob, receive the results of the votes together without ever knowing who voted for which option.

an analogy for FHE

ZKP



pip waits for bob. pip doesn't know the password for the lock, nor does he know which path bob will take, but as long as bob appears again and again, he knows for sure bob knows the password.

bob wants to meet pip. bob knows the password to the lock but he can't share it with pip. regardless of path taken, bob always shows up, proving bob knows the password.

an analogy for ZK proof

Motivation For The Work

Fully Homomorphic Encryption (FHE) and Zero-Knowledge Proofs (ZKPs) are computationally expensive.

Polynomial multiplication is a major bottleneck in these schemes.

TPUs/NPUs offer a potential solution for acceleration.

Polynomial Multiplication in Cryptography

- Why polynomials are important in cryptography:
  - Related to hard problems used in cryptography.
  - Effective way for computation representation.
- Polynomial operations:
  - Addition: Straightforward.
  - Multiplication: More complex, performance bottleneck.
  - Two common ways: Coefficient representation and value representation.

TPU Architecture and Suitability for Cryptography

- TPU overview:
  - Designed for AI workloads, especially matrix operations.
  - Systolic array architecture for high parallelism.
  - Low power consumption compared to CPUs and GPUs.
- Suitability for cryptography:
  - Potential for accelerating polynomial multiplication through matrix operations.
  - **Challenges**: Large coefficients and high polynomial degrees

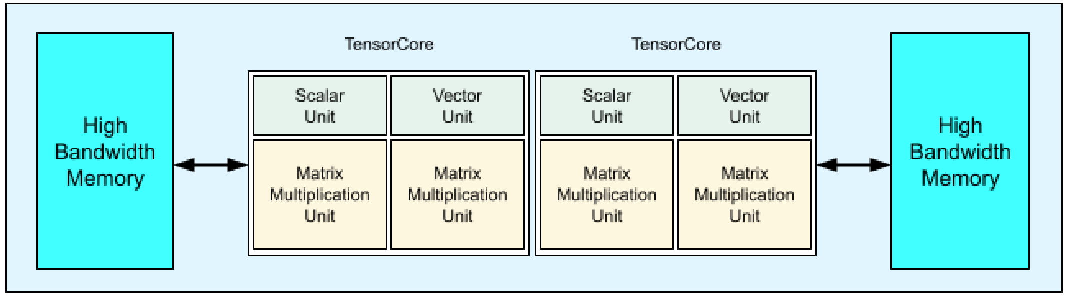


Image Source: Jouppi, Norman P., et al. "A domain-specific supercomputer for training deep neural networks." Communications of the ACM 63.7 (2020): 67-78.

Converting Polynomial Multiplication to Matrix Operations

- Incorporating modulo operation into multiplication.
- Converting to vector-matrix multiplication.
- Extending to matrix-matrix multiplication for multiple multiplications

A polynomial $f(x) \in \mathbb{Z}_q[x]/(x^n + 1)$ is in the form

$$f(x) = a_0x^0 + a_1x^1 + \cdots + a_{n-1}x^{n-1}, \tag{1}$$

$$(a_0, a_1, \ldots, a_{n-1}) \times \begin{bmatrix} b_0 & b_1 & \cdots & b_{n-1} \\ -b_{n-1} & b_0 & \cdots & b_{n-2} \\ \cdots & \cdots & \cdots & \cdots \\ -b_1 & -b_2 & \cdots & b_0 \end{bmatrix}. \tag{1}$$

Considering two degree-2 polynomials defined over $R_q = \mathbb{Z}_q[x]/(x^3 + 1)$, $a(x) = a_0x^0 + a_1x^1 + a_2x^2$ and $b(x) = b_0x^0 + b_1x^1 + b_2x^2$. The multiplication process can be converted to a vector-matrix multiplication operation

$$(a_0, a_1, \ldots, a_{n-1}) \times \begin{bmatrix} b_0 & b_1 & \cdots & b_{n-1} \\ -b_{n-1} & b_0 & \cdots & b_{n-2} \\ \cdots & \cdots & \cdots & \cdots \\ -b_1 & -b_2 & \cdots & b_0 \end{bmatrix}. \tag{1}$$

We can convert the multiplication into matrix multiplication format with moduli polynomial $x^n + 1$

Challenges and Solutions - Large Coefficients

- Challenge: TPUs are designed for smaller data types (e.g., bfloat16).
- Solution: Residue Number System (RNS)
  - Divide coefficients into smaller parts for parallel computation.
  - Independent instances can be executed on TPU without modification

Table 1: Common polynomial parameters for cryptographic schemes.

Scheme	Polynomial degree	Polynomial coefficients size
FHE (FV, BFV, CKKS)	2^{10} to 2^{14}	32 to 54 bits
PQC	2^{8} to 2^{10}	≤ 60 bits
ZKP (zkSNARK, zkSTARK)	2^{20} to 2^{21}	384 to 768 bits

Handling High Polynomial Degrees

- Calculate the product of sub-vectors and sub-matrices:

$$r_{1A} \leftarrow V_1 \times M_A \qquad r_{1B} \leftarrow V_1 \times M_B$$
$$r_{2C} \leftarrow V_2 \times M_C \qquad r_{2D} \leftarrow V_2 \times M_D$$

All results are vectors of dimension $n/2$.

- Calculate the first half of the final result $r_1 = r_{1A} + r_{2C}$, which is a vector of dimension $n/2$.
- Calculate the second half of the final result $r_2 = r_{1B} + r_{2D}$, which is a vector of dimension $n/2$.
- Concatenate r_1 and r_2 to form the final result, which is a vector of dimension n.

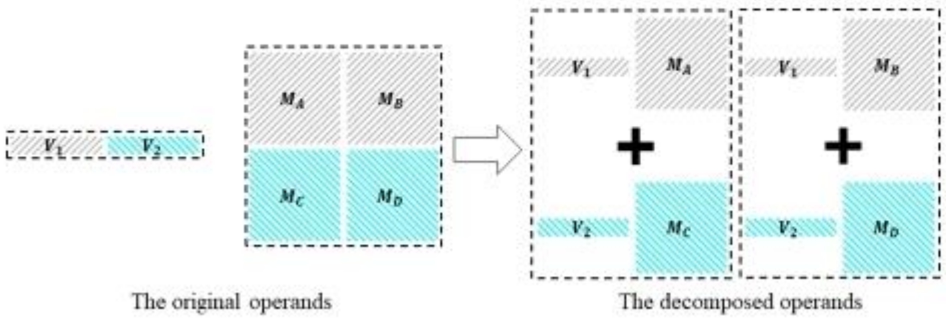


Figure 1: Demonstration of handling of polynomials with high degree. On the left side of the figure, we evenly break the vector into two sub-vectors and the matrix into four sub-matrices. The original vector-matrix multiplication is decomposed into four vector-matrix multiplications with smaller dimensions on the right side.

End to End workflow

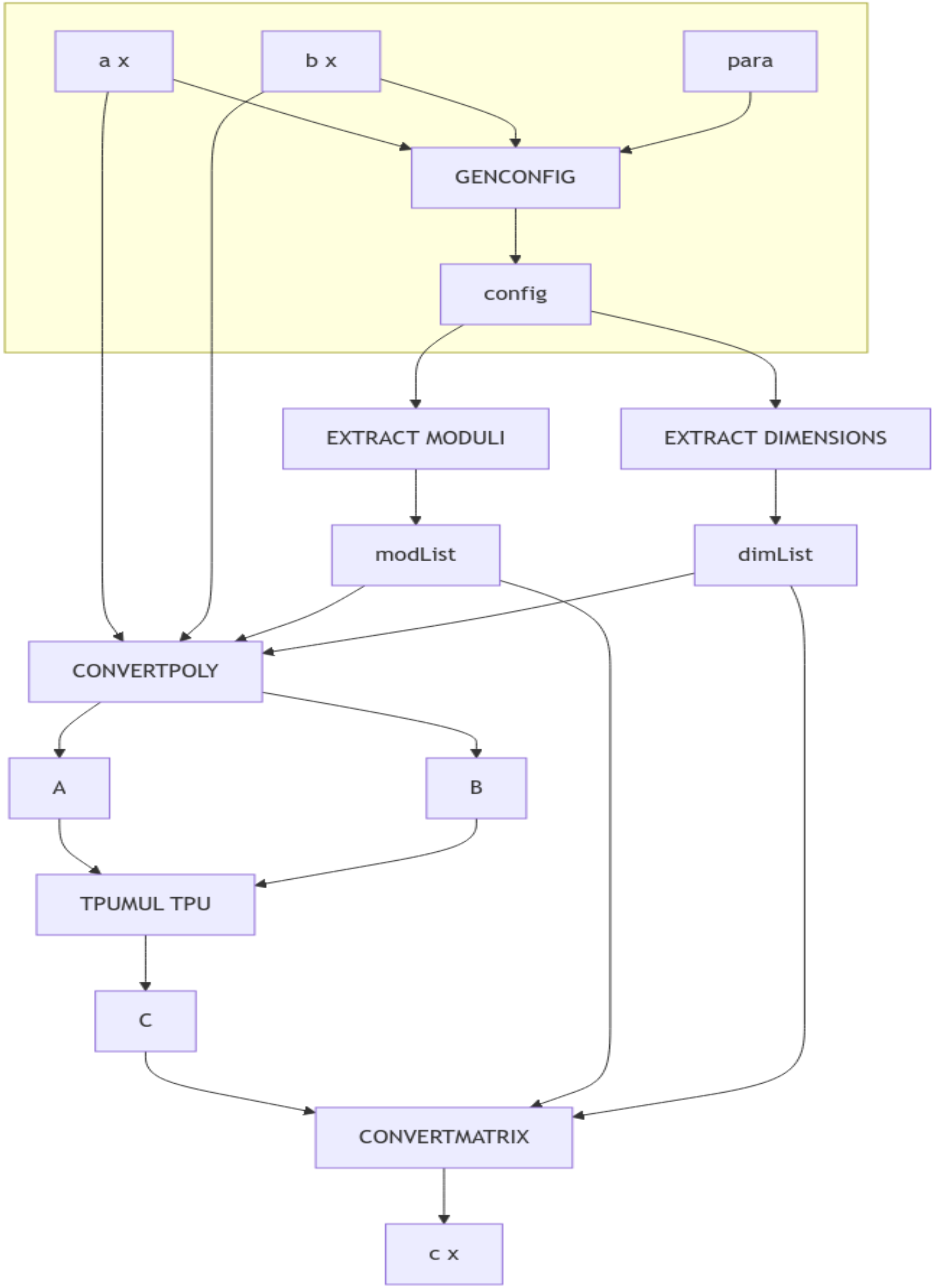
TPU Utilization: Leverage the matrix multiplication capabilities of TPUs to accelerate polynomial multiplication in cryptographic schemes.

Algorithm Overview:

•**Step 1:** Determine optimal parameters for decomposing polynomials based on the specific cryptographic scheme and TPU hardware constraints.

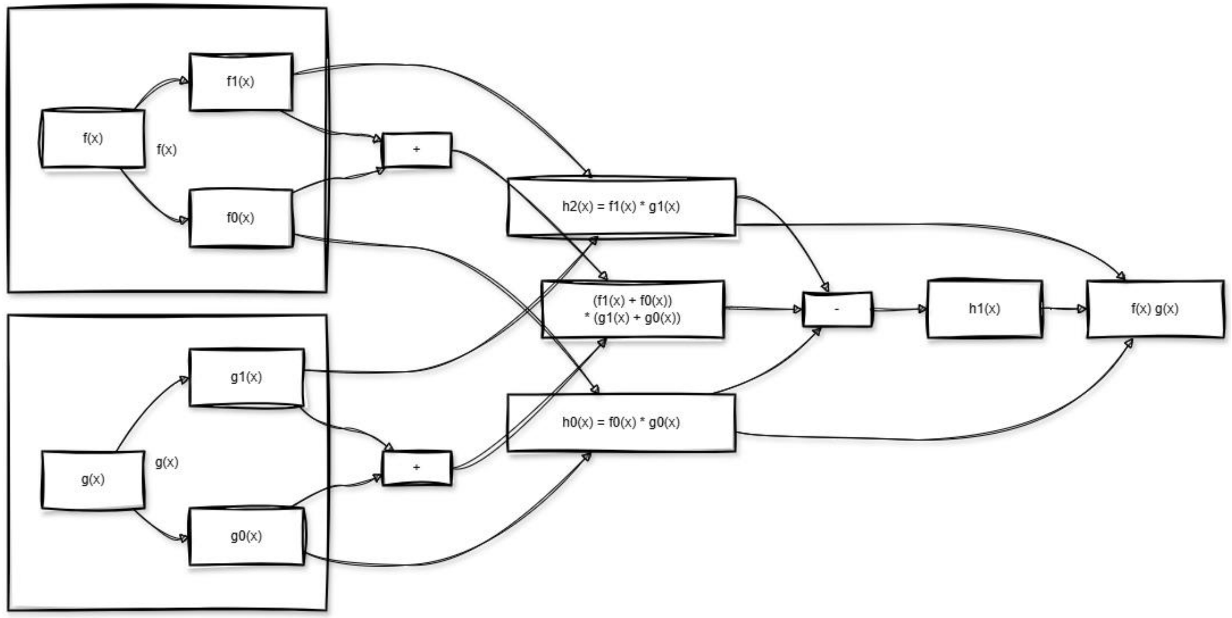
•**Step 2:** Convert polynomials to matrices and perform efficient matrix multiplication on the TPU.

•**Step 3:** Reconstruct the final polynomial product from the TPU's matrix multiplication results.



Future Work - Karatsuba Multiplication

- **Karatsuba Algorithm:** Reduces expensive multiplications by increasing additions/subtractions.
- **TPU Adaptation:** Decompose polynomials into smaller ones for TPU processing, leveraging its matrix multiplication strength.
- **Challenge:** TPUs are not optimized for efficient polynomial addition/subtraction, potentially requiring extra hardware and introducing overhead.

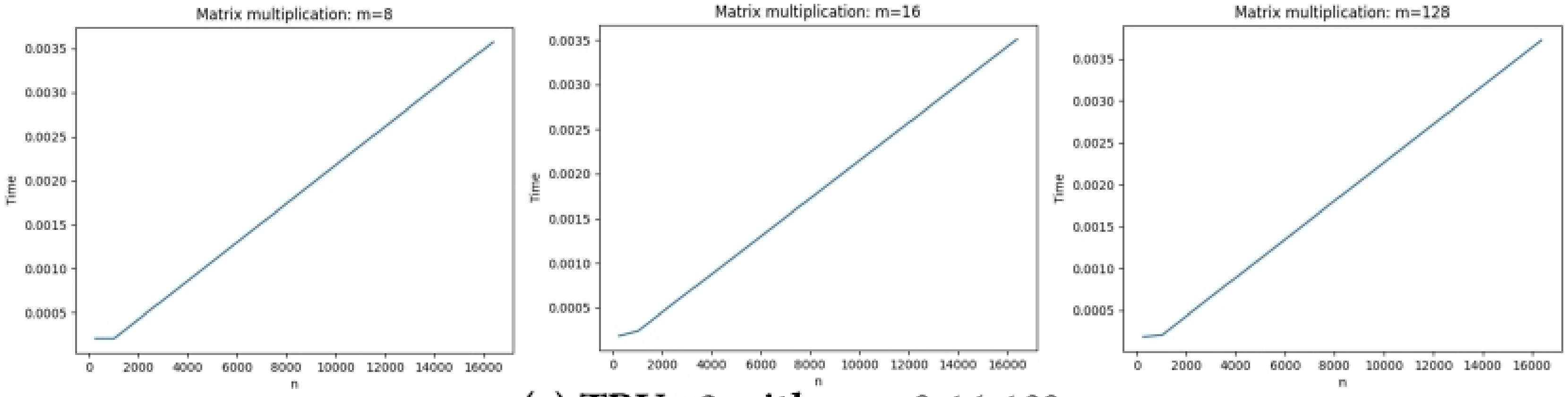


Karatsuba multiplication. The idea of Karatsuba multiplication [17] is trading one expensive multiplication with multiple cheap addition/subtraction operations. For two polynomials $f(x)$ and $g(x)$ with degree n, we can rewrite them as $f(x) = f_1(x)x^m + f_0(x)$ and $g(x) = g_1(x)x^m + g_0(x)$, where f_1, g_1 are polynomials of degree $n - m$, and f_0, g_0 are polynomials of degree $m - 1$. The product is then

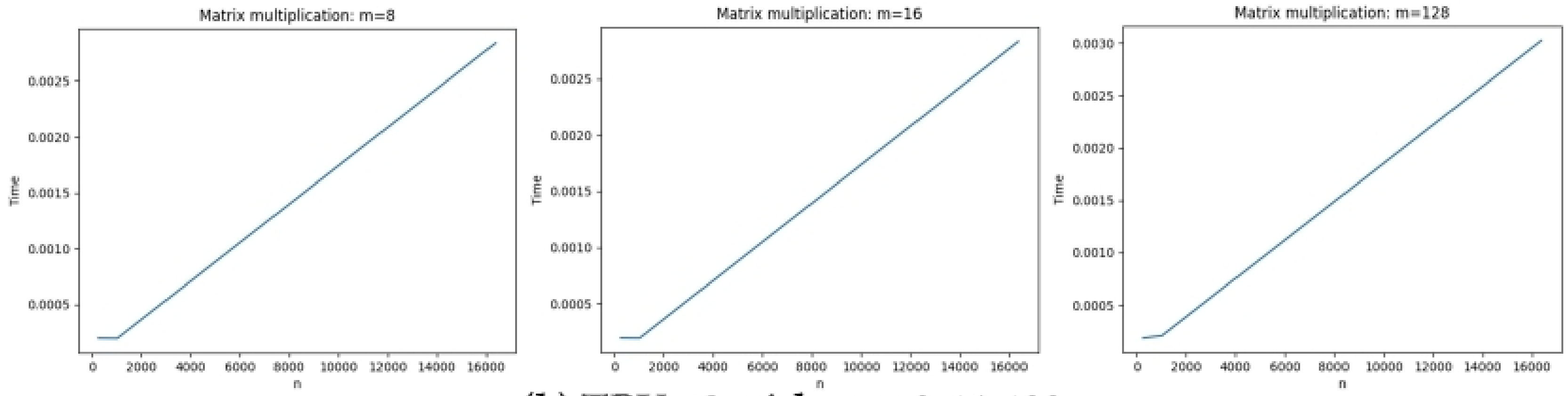
$$\begin{aligned} f(x)g(x) &= (f_1(x)x^m + f_0(x))(g_1(x)x^m + g_0(x)) \\ &= h_2(x)x^{2m} + h_1(x)x^m + h_0(x), \end{aligned}$$

where $h_2(x) = f_1(x)g_1(x)$, $h_1(x) = (f_1(x) + f_0(x))(g_1(x) + g_0(x)) - h_2(x) - h_0(x)$, $h_0(x) = f_0(x)g_0(x)$.

Experimental Results and Evaluation



(a) TPU v2 with $m = 8, 16, 128$



(b) TPU v3 with $m = 8, 16, 128$

Summary of experiment results using Google TPU with different configurations.

Conclusion

- TPUs show promise for accelerating polynomial multiplication in cryptography.
- RNS and divide-and-conquer address challenges of large coefficients and high degrees.
- Future work will focus on further optimizations and end-to-end scheme design.

Questions

Thank you!

Rabimba Karanjai
rabimba@cs.uh.edu

Read More!