Parallelization of Fully Homomorphic Data Encoding
Jess Woods¹, Ada Sedova², Oscar Hernandez¹

¹Computer Science Research Group, Oak Ridge National Laboratory  ²Biophysics Group, Oak Ridge National Laboratory

Introduction

- Data privacy is important for healthcare data, secure computation, etc.
- Homomorphic encoding scheme: supports operations on encoded data (Fig 1)

\[
\begin{align*}
    d_1 + d_2 &= d_1 + d_2 \\
    d_1 \cdot d_2 &= d_1 \cdot d_2
\end{align*}
\]

Figure 1: Homomorphic Operations

- Fully homomorphic scheme: supports arbitrary number of additions and multiplications [1]

Objective: Implement FHE scheme to allow user to outsource computation (the function f) to some “cloud” (separate user, supercomputer, etc.) without revealing any data (Fig 2)

\[
    f(d)
\]

Figure 2: Desired Model

Problems: time and memory
- Costly large integers used for security
- Costly “Recode” operation, used to mitigate noise growth (red in Fig 3)

Hypothesis: A parallelized implementation will mitigate data transformation costs and make fully homomorphic encoding feasible

Methods

Scheme used: Dijk et al’s fully homomorphic encoding scheme over the integers [2]
- Operations listed in Fig 3

Theoretical Improvements:
- Compressed public key size [3]
- Batched data (multiple bits per encoding) [4]

Implementation Improvements:
- GPU operations with CUDA
- Algorithm-level and OpenMP thread-level parallelism
- Big number handling with GMP library

Programming Languages:
- Python (proof of correctness sketch)
- C++ (better memory control)
- Julia (possible future of parallel computing)

Tested for:
- Summit (and future) supercomputers
- Smaller computing clusters
- Personal Laptop

Results

Our Contributions:
- First parallelization of Dijk et al’s scheme
- First to incorporate both the theoretical improvements
- Current fastest implementation of Dijk scheme
- Our comparison of languages/models used to help design and choose future supercomputer software

Conclusion

Future applications:
- Secure machine learning, achieved with homomorphic matrix multiplications
- Healthcare transactions
- Library functions that hide complex operations

With such a library and our achieved speed-ups, fully homomorphic encoding will be more feasible than ever for practical use by non-specialists.

---