cuECC

Accelerating Elliptic Curve Cryptography with GPUs

Minjae Gwon

Introduction

- Motivation
- Backgrounds

Method

- Design
- Implementations

Result

- Benchmarks
- Conclusion

POSTECH

Introduction

Motivation and Backgrounds

POSTECH

Motivation

Introduction

Significance of Asymmetric Cryptography

Plays a pivotal role in modern security systems and various applications.

Computational Challenges

Asymmetric cryptography is computationally intensive.

Enhancing Throughput of ECC by Leveraging Parallelism

Parallelize ECC computations within GPU powered by CUDA.

Backgrounds

Introduction

- **Asymmetric Cryptography**
  - Utilizes pairs of keys: public key and private key.
  - Public keys encrypt and verify, while private keys decrypt and sign.
  - Keys are mathematically related.
- **Elliptic Curve Cryptography (ECC)**
  - Asymmetric cryptography based on elliptic curves over finite fields.
  - Curve defined as $y^2 = x^3 + ax + b$.
  - secp256k1, secp256r1, etc.

Method

Design and Implementations

POSTECH

Design

Method

- Scope
  - Support only secp256k1 to relax the problem.
  - secp256k1 is widely used curve.
- Implementation Strategy
  - Kernels for each ECC operation in CUDA.
  - Parallel computation for multiple ECC points per kernel.
  - Python bindings using Foreign Function Interface (FFI).
  - Performance comparison with a pure-python implementation.

Implementations

Method

Uint256 Ops — Addition, Subtraction, Multiplication, etc.

Finite Field Ops — Modular, Power, Division, Inverse, etc.

Based on 256bit unsigned integer operations.

Elliptic Curve Ops — Addition, Subtraction, Multiplication, etc.

Based on Finite Field operations.

Implementations

Method

ECC Kernels

Public Key Generation

Specialized for secp256k1

Shared Library

Wrapping Kernels with Glue Codes written in C

Able to be compiled as shared library

Python Bindings

Foreign Function Interface for Python

Using built shared library and ctypes

Implementations

Method

- Reference Implementation
  - For checking accuracy and comparing performances.
  - Based on https://github.com/mohanson/cryptography-python.

Result

Benchmarks and Conclusion

POSTECH

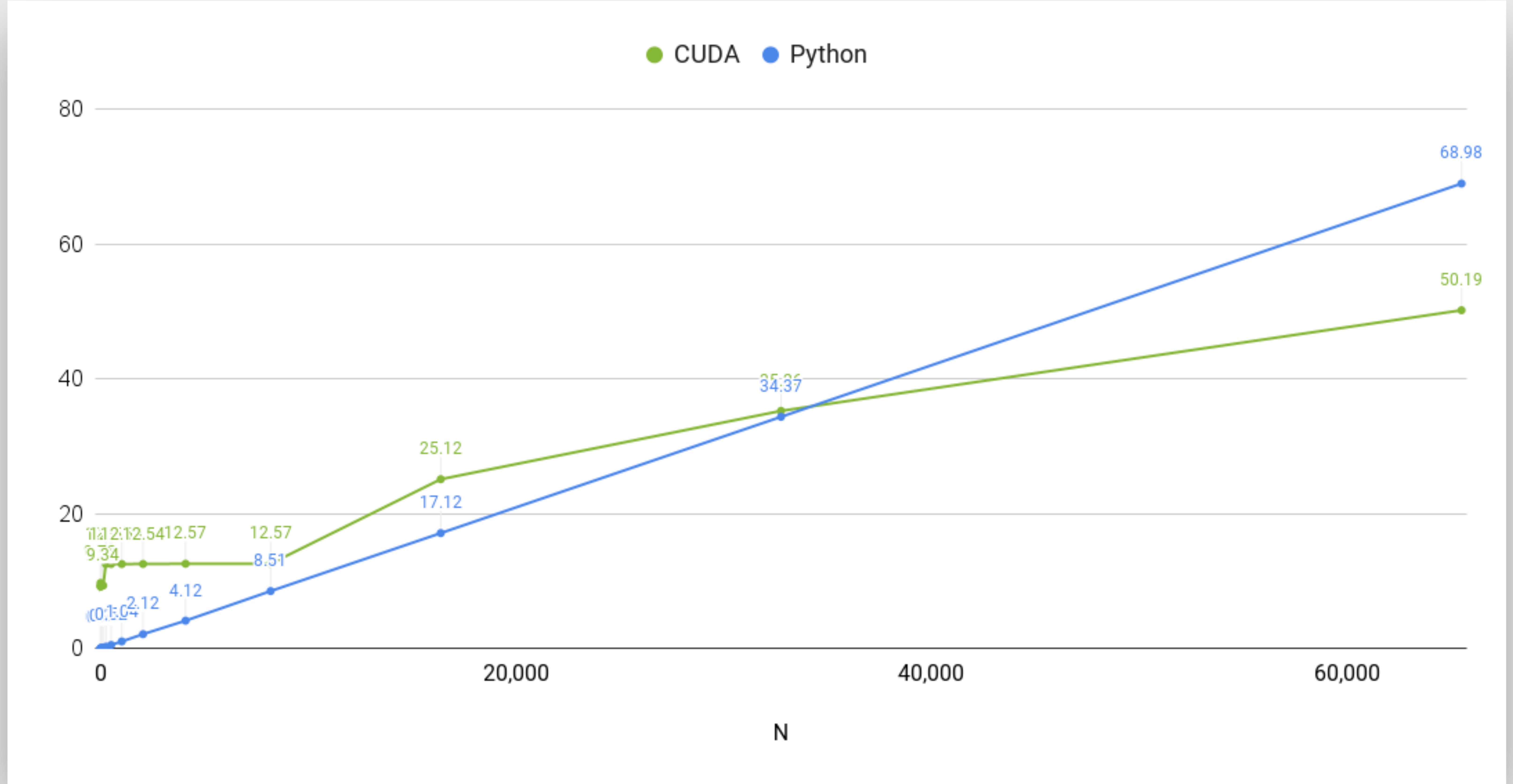
Benchmark – Public Key Generation

Result

- Comparison Scenario
  - Evaluate two implementations for 2^n private keys.
  - Randomly generated private keys.

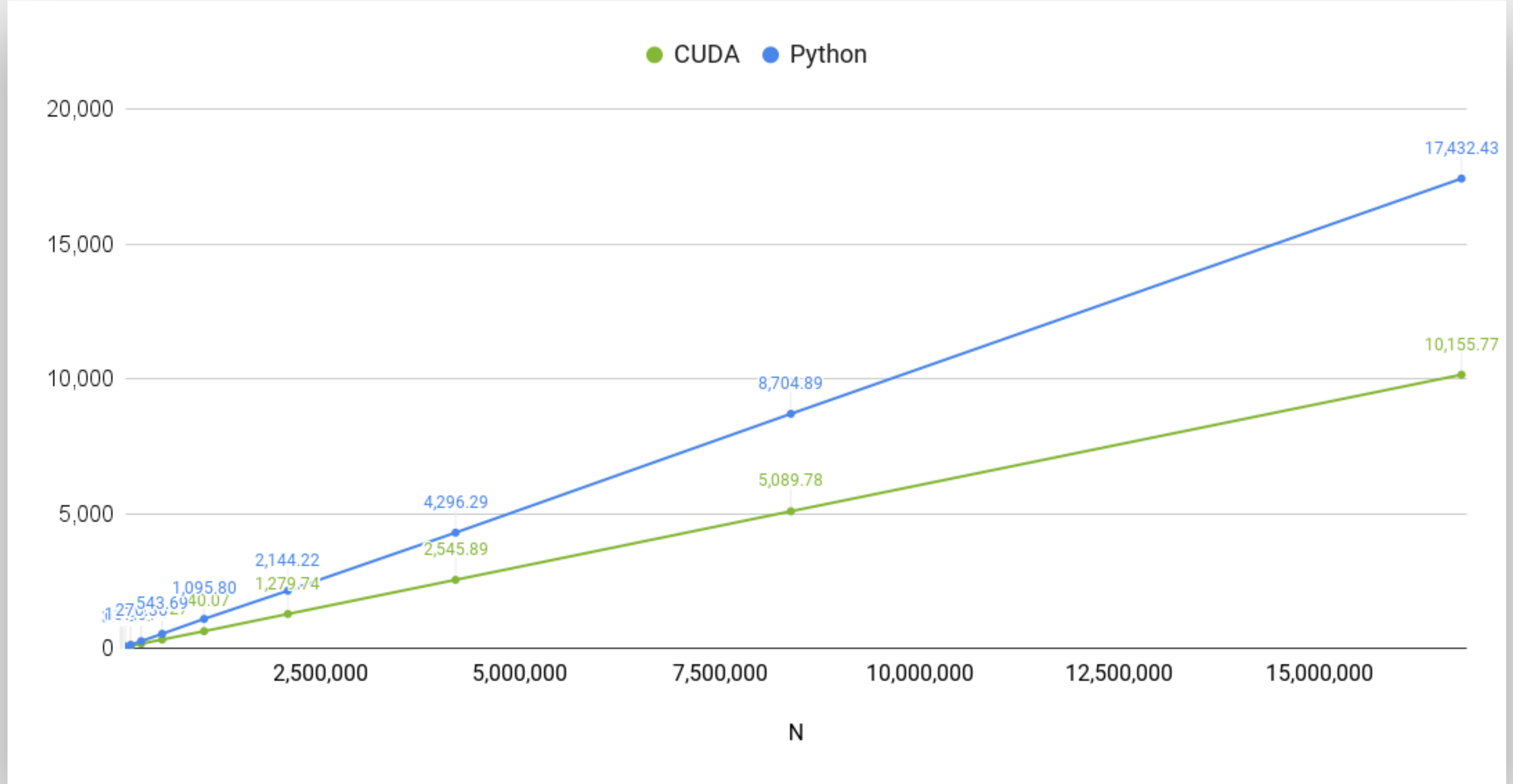
Benchmark – Public Key Generation

Result



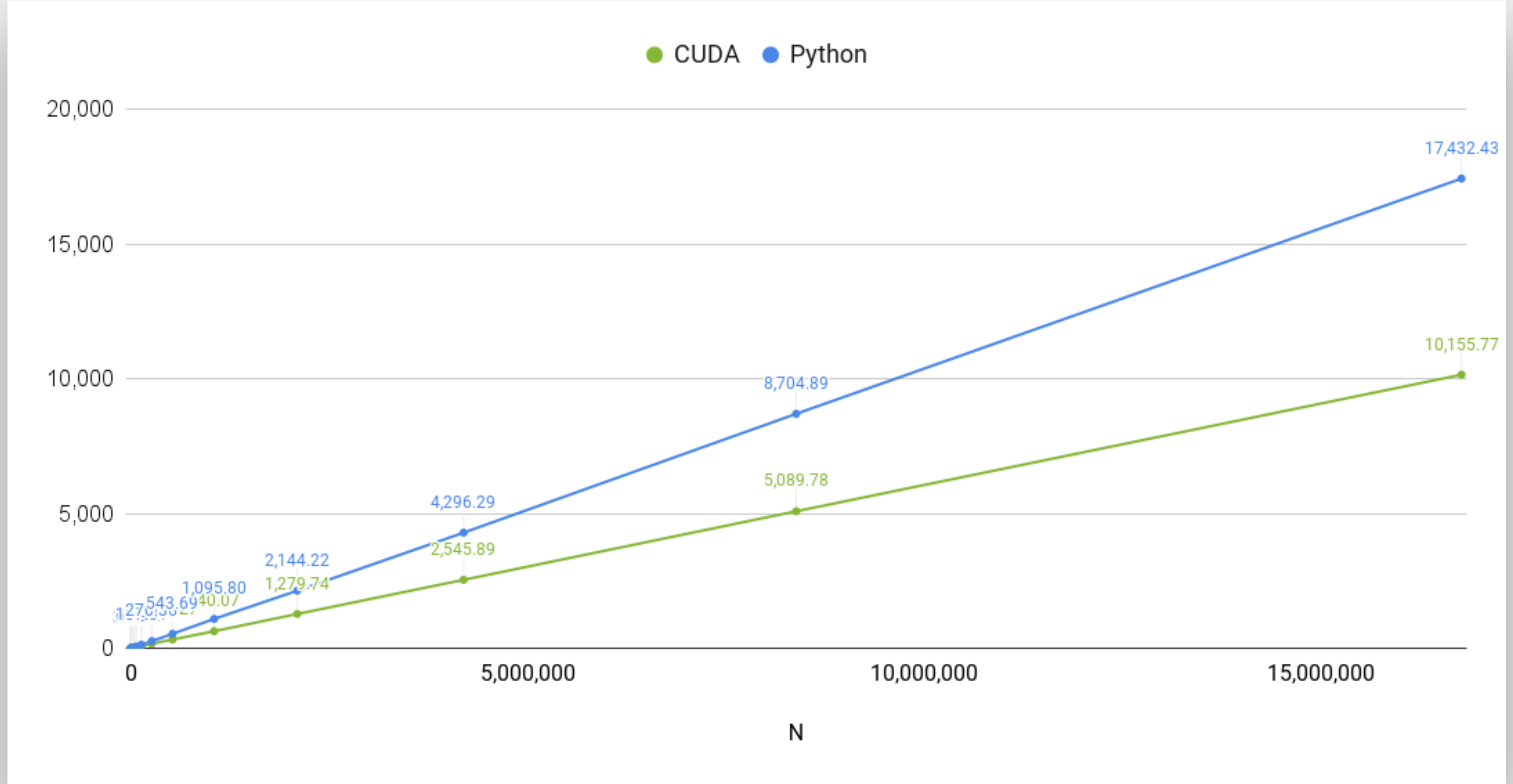
Benchmark – Public Key Generation

Result



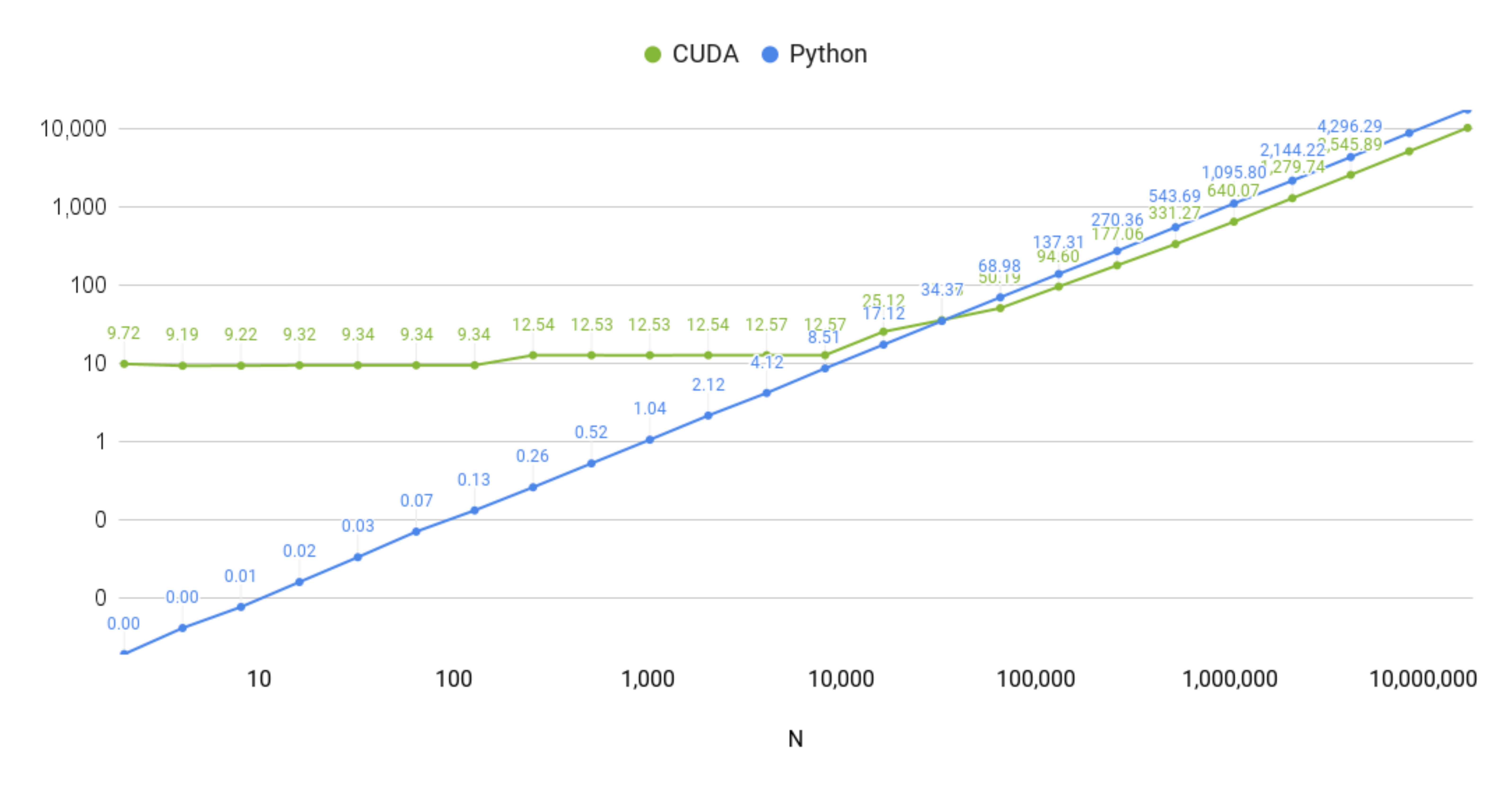
Benchmark – Public Key Generation

Result



Benchmark – Public Key Generation

Result



Conclusion

Result

- **GPU Advantage**
  - The project highlights the scalability potential of GPU-accelerated ECC.
  - Particularly in scenarios with a high volume of private keys.
- **Parallelism Challenges**
  - The lack of parallelism in curve point operations underscores an area for potential future enhancement.

cuECC

Accelerating Elliptic Curve Cryptography with GPUs

Gwon Minjae, Dept. of Computer Science & Engineering, POSTECH.

Appendix

https://github.com/betarixm/cuECC/

POSTECH