Kaining Zhong

kainingzhong@icloud.com | (447) 902-2049 | github.com/PRESIDENT810

EDUCATION

University of Illinois, Urbana Champaign - *Electrical & Computer Engineering, MEng.* Aug. 2022 – May. 2024

The Chinese University of Hong Kong - *Computer Science, BEng.* Aug. 2017 – May. 2021

SKILLS

Programming languages: C++, Python, Golang, Rust, TypeScript, Bash, Java, SQL
Dev Tools: AWS, Git, Neovim, GitHub Enterprise, JIRA, Confluence, VSCode
Skills: CMake, Bazel, LLVM, LLDB, Docker, CUDA, PyTorch, Apple MLX

Work Experience

Tesla - *Software Development Engineer Intern, Vehicle Software* Augest. 2023 – Dec. 2023

Mobile Engineering - Build Infra Docker, Xcode, React Native, GitHub Actions, Jenkins

- **Built GitHub workflows and Jenkins pipeline** for CI/CD automation with Docker containers & Mac minis as build machines. **Setup monitoring service using Prometheus and Grafana** to & visualize and analyze hardware utilization.
- Optimizing iOS & Android CI pipeline by implementing various caching strategy including a JS bundle injection, **reducing build time from 45 min to less than 15 min**.

Amazon - *Software Development Engineer Intern, AWS CloudFront* May. 2023 – August. 2023

Control Plane - Realtime Logging Java, C++, AWS Kinesis, AWS EC2, AWS CloudWatch

- Developed, tested and deployed an integral component of AWS CloudFront's realtime metrics monitoring system, focusing on **leveraging Google FlatBuffers to serialize realtime metrics** collected from CloudFront's Nginx & Squid caching servers, and **utilizing zstd to compress serialized binary byte stream**, which reduced 48% size of serialized data stored on Kinesis
- Employed AWS C++ SDK to feed data on realtime metrics processing cluster io **AWS Kinesis streaming system**, and employed AWS Java SDK to consume realtime metrics and display it on **AWS CloudWatch monitoring service**.

TikTok - *Software Development Engineer, iOS Infrastructure* July. 2021 – July. 2022

LLVM Compiler Infrastructure & Xcode Toolchain C++, Xcode, FaaS, Object Storage

- Improved Xcode debugging experience by optimizing Apple's LLDB, and deployed Xcode toolchain bundled with customized LLDB to multiple iOS APP's development team, **reducing initialization time for debugging from 200s to 30s**
- Fixed bugs for LLVM's LLD linker and migrated from Apple's LD64 linker to LLD linker, enabling faster linking for iOS applications including Lark & Toutiao; **reducing linking time from 120s to 60s in average**.
- Implemented a LLDB performance analyzing tool by dynamically rebinding exposed C++ Script Bridge API symbols in LLDB bundled in Xcode, reporting statistics such as time for loading images, variable evaluation, etc.
- Maintained a backend service implemented by **ByteFaaS serverless** that managed Xcode toolchains compiled from internal fork of Apple's Swift repository, using TOS as Object Storage Service to store toolchains.

Open Source

- **LLVM Project**: reported issues and fixed bugs, including thin archive support in LLDB, and Dtrace support in LLD.
- **Apple MLX** Implemented C++ layer's Diagonal operator with the frontend Python API.
- **YCSB**: implemented a etcd binding using jetcd to benchmark etcd's CRUD performance.

Projects

Needle, a PyTorch-like deep learning framework Python, C++, CUDA

- Implemented a PyTorch style deep learning framework with components including **dynamic computation graph, backward automatic differentiation, dataloader, optimizer, some widely used neural network modules, and backend operators including a im2col optimized convolution operator supported by C++ and CUDA**.
- Trained ResNet9 model on CIFAR-10 dataset, and LSTM model on Penn Treebank dataset entirely relying on Needle framework.

Bustub, a Relational Disk Based Database C++, GTest, SQL, Relational DBMS

- Implemented a **concurrent B+ tree index** for faster data retrieval, which supports high concurrency access using latch crabbing to avoid unnecessary multi-thread synchronization, along with a **buffer pool manager** using **extendible hash table** and **LRU replacement policy**.
- Implemented **volcano model operators** that execute SQL query plans, supporting a subset of SQL syntax including sorting, joining and aggregation.
- Implemented a **transaction manager to support concurrent SQL queries**, which supports isolation levels up to Repeatable Read, along with a **deadlock detector** that aborts transaction when deadlocks are detected.

Distributed, Fault-tolerant KV NoSQL Database Golang, Raft, RPC

- Built a fault-tolerant Key-Value NoSQL database based on Raft consensus algorithm using Golang's concurrency & RPC packages, supporting Put, Get, Append operations, and guaranteeing Consistency & Partition tolerance.
- Implemented optimizations including fast log recovery and Snapshot persistence mechanism.