Ningqi Wang

+1-412-567-7252
nq.maigre@gmail.com
in /ningqi-wang
/oxf3cd

Education

Carnegie Mellon University | *M.S. in Information Networking; GPA: 3.50/4.00* Pittsburgh, PA | *Aug. 2020 – Dec. 2021*

- Teaching Assistant for course 17-681 Java for Application Programmers in Spring 2021 and Fall 2021

Tongji University | *B.E. in Computer Science and Technology; GPA: 91.37/100* Shanghai, China | *Sept. 2015 – Jul. 2020*

- Graduated with Honor: Shanghai Outstanding Graduate Award
- Tongji Excellent Undergraduate Scholarship in 2017 and 2018 (Ranking 14/177 and 1/164 respectively)

Experience

Apple | *GPU System Software Engineer Intern* Pittsburgh, PA | *May. 2021 – Aug. 2021*

- Established a color transform pipeline for virtual graphics within the hypervisor, where render requests in guest OS would be captured and then be passed along the pipeline to physical graphics on host
- Optimized internal APIs to reduce the overhead of data transfer and developed GPU shaders that efficiently applied color transforms to pixel data, powering visual accessibility features

Autodesk | *Software Engineer Intern* Shanghai, China | *Sept. 2019 – Feb. 2020*

- Replaced MSVC with LLVM compiler toolchain for AutoCAD on Windows (consisting of 100k-level lines of C/C++ codes)
- Upgraded internal CI/CD platform to enable test automation on AutoCAD builds generated by LLVM on Windows
- Developed Clang plugins to analyze dependency of source files and detect redundant codes

Selected Projects

Parallel Algorithm for Fast Distributed Training | *PyTorch, RPC*

- Proposed a synchronous mechanism (i.e. LOSP) based on Parameter Server (PS) to accelerate neural network training on a distributed GPU cluster, where computation and communication processes were overlapped to reduce synchronization overhead
- Implemented LOSP and other distributed training algorithms, and developed scripts to automatically conduct comparison experiments on different neural network architectures (CNN, AlexNet, LSTM, ResNet, and DenseNet) in various cluster size (4/8/16) and hyperparameter settings
- Achieved ~50% shorter convergence time in most cases (with same or better performance), ~0 GPU idle time, and significantly less communication cost, with Local-SGD as the baseline algorithm
- Paper published: H. Wang, Z. Qu, S. Guo, N. Wang, R. Li and W. Zhuang, ”LOSP: Overlap Synchronization Parallel With Local Compensation for Fast Distributed Training,” in IEEE Journal on Selected Areas in Communications, vol. 39, no. 8, pp. 2541-2557, Aug. 2021, doi: 10.1109/JSAC.2021.3087272.

Cloud Computing Projects | *MapReduce, Spark, MySQL, HBase, Docker, Kubernetes*

- Established ETL pipelines to process 1.4TB tweets' data and analyze similarity between users based on users' interactions
- Designed and constructed high-performing, load-balanced, and fault tolerant cloud services that supported block chain verification requests and complex DB queries, with limited budget and strict performance requirements
- Improved throughput on HBase by 2531.33%, by optimizing rowkey's design; Achieved 6.595x higher throughput on MySQL by optimizing tables' design to reduce data redundancy and customizing SQL statements to minimize query overhead

Computer System and Infrastructure Practices | *C, C++, Golang, Node.js*

- **Real-time OS Kernel** - a basic kernel with multi-threading, preemption, context switching, priority schedule, and mutex
- **Raft Consensus Algorithm** - one implemented with Golang, supporting arbitrary binary data
- **Distributed File System** - one implemented with read-write locks and supporting single writer and multiple readers
- **Compiler** - a series of software that supported:
  1. creating configurable LL(1) or SLR(1) parser based on input (i.e. parser generator)
  2. parsing a C-like language (supporting function calls)
  3. applying processor-independent optimizations and peephole optimization on intermediate representations
  4. generating executable assembly codes on x86 processors
- **Micro-Computer on FPGA** - a micro-computer with hardware and software components, including:
  1. a MIPS32 CPU with 5-stage pipeline and cache, and Coprocessor 0
  2. controllers and adapters for DDR2 SDRAM (main memory), SD card (external storage), Ethernet card, and VGA screen
  3. a game where players used another computer (connected by cable) to control the game via Ethernet frames
  4. a bootloader to initialize peripherals and load the game from SD card to main memory

Skills

Languages	C, C++, Objective-C, Golang, Java, Scala, Python, R, JavaScript, TypeScript, Verilog
System Software	CMake, Make, LLVM, GDB, perf, Valgrind
Web Full Stack	React, Node.js, Gofiber, Vert.x, Django, Nginx, Kafka, MySQL, NoSQL, Redis, Docker, Kubernetes
Big Data & Machine Learning	Terraform, Spark, MapReduce, HBase, HDFS, pandas, NumPy, scikit-learn, PyTorch, TensorFlow