Jonathan Jiang

jonathanrsjiang@icloud.com linkedin.com/in/jonathanrsjiang +1-647-408-3030

Skills

Languages: C++, Python, Java, JavaScript, Golang, Lua, SQL, HTML, CSS, Rust

Technologies: Linux, Git, Hadoop, Hive, Spark, NumPy, Jenkins, React, LLVM, PostgreSQL, Node.js, Next.js

Experience

Huawei Technologies Toronto, ON

Software Engineer Intern (Big Data Platform) *May 2023 - May 2024 · 1 yr*

Hybrid Execution SQL Query Engine (Java, C++):

* Assisted in the translation of critical classes to C++ and leveraged JNI for integration with legacy Java components.
* Spearheaded effort to create and refine a shared development environment to speed onboarding process for developers.
* Engaged in code reviews, inspections, and contributed documentation to preserve code quality and maintainability.
* Conducted thorough testing and debugging to ensure the reliability and stability of implemented modifications.

Benchmarking and Profiling Pipeline (Python):

* Piloted benchmarking and profiling toolkit to greatly improve developer efficiency for end-to-end result analysis.
* Centralized results distributed across Spark cluster nodes and extracted metrics including hotspots and operator usage.
* Automated generation of flamegraphs and ensured interoperability with profilers including perf, async, and VTune.
* Built logging system to store and compare different builds to identify changes in performance across operator functions.

OLAP SQL Operator Optimization (C++):

* Upgraded SQL Query hash aggregation and join operators by incorporating novel data structures and hashing functions.
* Iterated through all combinations of changes using A/B benchmarking across varying scales to identify best candidates.
* Refactored code to optimize performance using software design techniques including SIMD, vectorization, JIT, and caching.
* Improved operator performance by 19%, and contributed 7% end-to-end, measured by the TPC-DS dataset benchmark.

FPGA Decoding Support (Java):

* Developed tools for examination of Apache ORC file format encoding and decoding to explore potential optimizations.
* Modified run-length encoding writer and reader behaviour logic to meet FPGA specifications involving sub-encodings.
* Experimented with storage-level optimizations of ORC stripe and footer composition to maximize performance of FPGA.
* Facilitated Xilinx FPGA integer decoding speed 500% faster than standalone server CPU.

Data Cleaning Pipeline for MoE pre-training (Python):

* Designed robust training data cleaning pipelines for individual subject experts in a Mixture of Experts model.
* Automated data pre-processing tasks, including normalization, missing value imputation, and outlier detection.
* Reviewed research papers and evaluated implementations of training optimizations including curriculum learning.
* Automated hill-climbing of filtering parameters for RedPajamaV2 and CCnet datasets using judge LLMs and ablations.

Projects

Counter Strike: Arbitrage (JavaScript, Python):

- Developed Python scripts to scrape third-party CS:GO item exchanges to identify arbitrage opportunities using REST API calls.
- Utilized heuristics such as trade volume, previous sales, transaction costs and others to evaluate relative risk of items.
- Traded $500 CAD of initial capital to $12000 CAD using the script to select transactions until opportunities diminished.

Rotator Cuff Rehabilitation Device (Arduino, Python):

- Prototyped at-home rehabilitation device used to facilitate remote monitoring of post-surgery physiotherapy of the shoulder.
- Implemented calibration and signal fusion algorithms including Kalman filters to accurately evaluate measurements of sensors.
- Integrated muscle repetition detection and recovery projection functions into result graphs with NumPy and Matplotlib.
- Designed application UI, and implemented account system to store and share patient records.

Education

McMaster University Hamilton, ON

Bachelor of Engineering, Software and Biomedical Engineering Co-op, President's Award of Distinction *Sept 2020 - Present*

Personal Interests

```
def fishing():
    while True:  eat(); sleep(); fish()
```