CHEN ZHONG

(682)256-0650 zhongch4all@gmail.com linkedin/zhongch4g zhongch4g.github.io

SUMMARY

I am a computer science Ph.D. candidate at the University of Texas at Arlington. My research projects include implementing caching, indexing and key-value storage systems in user and kernel space. I worked on building a framework that unifies existing state-of-the-art index designs with Tencent America database team. I am currently collaborating with VMWare vSAN team to enable applications to amortize I/O interfacing overhead for Key-Value operations.

EDUCATION

University of Texas at Arlington | *MEng; Ph.D. candidate in Computer Science* **Sep. 2019 – Present**

Beijing University of Posts and Telecommunications | *MEng, in Software Engineering* **Sep. 2016 – Jul. 2019**

Jiangxi University of Science and Technology | *BEng, in Software Engineering* **Sep. 2012 – Jul. 2016**

WORK EXPERIENCE

Tencent America LLC. **Aug. 2022 – Dec. 2022**

Database Research and Development Intern | C/C++, In-memory/on-disk index, Database *Bellevue, WA*

- Explored state-of-the-art approaches in database management system and integrated them into production code.
- Designed and developed a novel extensible index spanning memory and disk for next generation database systems. The performance of this system is shown to result in 30X increase in throughput in various benchmarks.

JD.com, Inc. **Aug. 2017 – Mar. 2018**

Research and Development Intern | Python, Hive, Data analysis *Beijing, China*

- Analyzed user behaviour data to extract mission critical features to develop a prediction model which is integrated into back-end services of JD.com Analytics team.
- Developed an user analysis module for precision marketing to deliver relevant ads according to each user's interest that optimized the click-through rate from 2% of the base to 38%.

PUBLICATIONS

- **Chen Zhong**, Qingqing Zhou, Yuxing Chen, Xingsheng Zhao, Kuang He, Anqun Pan, Song Jiang, "IndeXY: A Framework for Constructing Extensible Large Indexes for OLTP Databases" (ICDE '24).
- Xingsheng Zhao, Prajwal Challa, **Chen Zhong**, and Song Jiang, "Developing Index Structures in Persistent Memory Using Spot-on Optimizations with DRAM" (ICPE 2024).
- Sujit Maharjan, Shuaihua Zhao, **Chen Zhong**, and Song Jiang, "From LeanStore to LearnedStore: Using a Learned Index to Improve Database Index Search" (HDIS 2023, ***Best Paper Award***).
- Xingsheng Zhao, **Chen Zhong**, Song Jiang, "TurboHash: A Hash Table for Key-value Store on Persistent Memory", (SYSTOR '23).
- **Chen Zhong**, Prajwal Challa, Xingsheng Zhao, Song Jiang. "Buffered Hash Table: Leveraging DRAM to Enhance Hash Indexes in the Persistent Memory" (NVMSA '22, ***Best Paper Candidate***).
- **Chen Zhong**, Xingsheng Zhao, Song Jiang, "LIRS2: An Improved LIRS Replacement Algorithm" (SYSTOR '21).

RESEARCH & PROJECT EXPERIENCE

Efficient Access of Distributed Key-value Storage | *C/C++, Linux kernel* **2023 – present**

- Characterizing performance bottlenecks within the KV storage stack, conducting in-depth analyses to amortize individual KV access overheads.
- Exploring the tradeoff between write latency and data persistency, while implementing strategies for managing KV cache.
- Collaboration with **VMware**. Keywords include **SPDK, RocksDB, Caching, NVMe, KVSSD**.

A Memory-disk-spanning Index Design | *C/C++, Python, Shell* **2022 – 2023**

- Implemented a framework with well-designed mechanisms and policies to glue a selected existing in-memory index (Index X) and an existing on-disk index (Index Y) into one extensible index (IndeXY)
- Built performance-critical capabilities to the index, including identifying hot/cold data, granularity selection and selective unloading of data to the disk.
- Collaborated with **Tencent America**. Keywords include **Indexing, Caching, RocksDB, B+ Tree**.

Improve Persistent Memory Hash Table Efficiency | *C/C++* **2021 – 2022**

- Designed a novel persistent memory hash table data structure that can efficiently take advantage of persistent memory hardware properties for improved performance.

A Cache Replacement Algorithm | *C/C++, Block/page strategies* **2020 – 2021**

- Incorporated a new data locality measure into the state of the art LIRS cache replacement algorithm that improve its performance by 19.1% across various workloads, with lower time and space overheads.