Jingji Chen

Personal Data

Phone: +1 (323) 447-1406
email: jingji.chen.000@gmail.com/chen3385@purdue.edu
Homepage: https://amadeuschan.github.io

Education

Starting from 2022 Aug.	Ph.D. in Computer Science (transferred from USC) Advisor: Prof. Xuehai Qian Department of Computer Science **Purdue University**, West Lafayette, USA
2019 Aug.-2022 Aug.	Ph.D. in Computer Engineering (passed the screening exam) Advisor: Prof. Xuehai Qian Ming Hsieh Department of Electrical and Computer Engineering **University of Southern California**, Los Angeles, USA GPA: 3.96/4.0
2015 Sept.-2019 July	Bachelor of Engineering Department of Computer Science and Technology **Tsinghua University**, Beijing, China Last-two-year GPA: 3.93/4.0

Experience

May 2022 - Aug. 2022	Research Intern at Microsoft Research, Redmond, WA (the RiSE lab, supervisor: Saeed Maleki)
July 2018 - Sept. 2018	Summer Research Intern at USC, Los Angeles, CA (supervisor: Xuehai Qian)

Research Interests

Machine learning systems, Graph processing systems and architectures,
Distributed systems, High-performance computing.

Preprints

J. Chen, Z. Chen, X. Qian, GNNPipe: Scaling Deep GNN Training with Pipelined Model Parallelism (arXiv 2023, link).

Publications or Accepted Papers

J. Chen, X. Qian, Khuzdul: Efficient and Scalable Distributed Graph Pattern Mining Engine. The 28nd Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'2023).

J. Chen, X. Qian, DecoMine: A Compilation-based Graph Pattern Mining System with Pattern Decomposition. The 28nd Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'2023).

G. Rao, J. Chen, J. Yik, X. Qian, SparseCore: Stream ISA and Processor Specialization for Sparse Computation. The 27nd Conference on Architectural Support for Programming

Languages and Operating Systems (ASPLOS'2022).

Y. Zhuo*, J. Chen* (* equal contribution), G. Rao, Q. Luo, Y. Wang, H. Yang, D. Qian, X. Qian, Distributed Graph Processing System and Processing-In-Memory Architecture with Precise Loop-Carried Dependency Guarantee. ACM Transactions on Computer Systems (TOCS), Volume 37, Issue 1-4, Artical No. 5, 2021.

Y. Zhuo*, J. Chen* (* equal contribution), Q. Luo, Y. Wang, H. Yang, D. Qian, X. Qian, SympleGraph: Distributed Graph Processing with Precise Loop-carried Dependency Guarantee. The 41st ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI'2020).

Skills

Language	Chinese (native)
	English (TOEFL:104(R30,L27,S22,W25), GRE(153V+169Q, 3.5AW))
Programming	Familiar with C/C++, Python, Java, MATLAB
	Basic knowledge about VHDL, SQL, C#, Go, JavaScript, etc.
	A fast learner in new programming languages and algorithms.
Tools/Frameworks	Tensorflow, PyTorch, Keras, Git, LaTeX, Numpy, Scipy, Lex&Yacc, MPI, openMP, clang-llvm, CUDA, cuDNN, etc.

Honors and Awards

Aug. 2019	Annenberg Fellowship, University of Southern California
Oct. 2018	Award for **Academic Excellence**, Tsinghua U
	Hengda Scholarship
	Scholarship for Technology Innovation and Excellence, Tsinghua U
Apr. 2018	**Outstanding Winner**(Highest award, 16/10000), ICM Contest, COMAP
	Xiao Shu-tie Scholarship for Applied Mathematics, In award of ICM
Oct. 2017	Scholarship for Academic Progress, Tsinghua U
May. 2017	3rd Prize, Hua-Luogeng Mathematical Modeling Contest, Tsinghua U
Aug. 2014	**Silver Medalist, National Olympiad in Informatics, China**

Teaching Activities

Spring 2021	Teaching Assistant of USC EE-451 (Parallel and Distributed Computation)

Course Projects

Dec. 2017	A THCO-MIPS CPU on FPGA
Nov. 2017	*Team Leader, Computer Organization Course Project*
	Designed a THCO-MIPS 16-bit CPU in VHDL with 5-stage pipeline. External devices available (PS2 keyboard, VGA monitor, etc.); (The Github Repo)
	In charged of the overall architecture design and several modules (controller & hazard detectors)

Mentored Undergraduate Students

Sept. 2021 - Aug. 2022	Zhuoming Chen (Tsinghua, currently a Ph.D. student@CMU)
June 2021 - Sept. 2021	Jingjia Luo (Tsinghua, currently a Ph.D. student@Tsinghua)
June 2020 - Sept. 2020	Dingyuan Cao (Tsinghua, currently a Ph.D. student@UIUC)
June 2020 - Sept. 2020	Sean Syed (University of Southern California)

Research Projects

Aug. 2023 – Sept. 2021 | GNNPipe: Scaling Deep GNN Training with Pipelined Model Parallelism (paper in submission, first author)

Advisor: Prof. Xuehai Qian

Existing distributed GNN training systems failed to efficiently support deep GNN models with far more than 2 layers – they exhibited poor weak scalability w.r.t. model depths. For example, training a 128-layer GNN with 16 GPUs could be $5.7\times$ slower than training a 16-layer model with 2 GPUs (with the same depth-to-GPU ratio) and costs $8.9\times$ more per-GPU communication traffic. It is due to the high communication complexity of existing distributed data-parallelism method. To tackle the issue, we proposed GNNPipe, the first distributed GNN system that targets deep GNN models and achieves a lower communication complexity with a novel chunk-based pipelined model parallelism method. The system is implemented in C++ on top of CUDA, cuDNN, cuBLAS, cuSPARSE and NCCL **(with roughly 16K lines of code)**. It outperforms our baseline by up to **2.45x** with up to **27.21x** less communication overhead. It also significantly outperforms DGL by one order of magnitude.

Aug. 2021 – July 2020 | SparseCore: Stream ISA and Processor Specialization for Sparse Computation (paper accepted by ASPLOS'22, second author)

Advisor: Prof. Xuehai Qian

Computation on sparse data is becoming increasingly important for many applications. Recent sparse computation accelerators are designed for specific algorithm/application, making them inflexible with software optimizations. To this end, we propose SparseCore, the first general-purpose processor extension for sparse computation that can flexibly accelerate complex code patterns and fast-evolving algorithms. We extend the instruction set architecture (ISA) to make stream or sparse vector first-class citizens, and develop efficient architectural components to support the stream ISA. The novel ISA extension intrinsically operates on streams, realizing both efficient data movement and computation. The simulation results show that SparseCore achieves significant speedups for sparse tensor computation and graph pattern computation.

Aug. 2021 – June 2020 | Khuzdul: An Efficient and Scalable Distributed Graph Pattern Mining Engine (paper accepted by ASPLOS'23, first author)

Advisor: Prof. Xuehai Qian

We designed and implemented a distributed engine targeting efficient large-scale graph pattern mining (GPM) **(with roughly 5000 lines of C++ code)**. Specifically, we propose: 1) a fine-grained domain-knowledge-aware abstraction named extendable embedding that efficiently decouples graph pattern mining algorithms and their distributed execution so that our engine can seamlessly enables distributed execution of existing non-scalable systems like Automine with low computation and communication overhead; 2) a memory-effective DFS-BFS hybrid scheduler providing sufficient parallelism for communication reduction techniques; 3) a set of communication and computation optimizations enabled by the new abstraction and novel scheduling. As a result, systems built upon our engine significantly outperforms state-of-the-art distributed systems and scales to large graphs with more than one hundred billion edges, a new milestone for large-scale graph pattern mining.

Aug. 2020 – Aug. 2019 | DecoMine: A Compilation-based Graph Pattern Mining System with Pattern Decomposition (paper accepted by ASPLOS'23, first author)

Advisor: Prof. Xuehai Qian

We designed and implemented a high-performance general-purpose graph pattern mining (GPM) system **(with roughly 10000 lines of C++ code)** based on pattern decomposition to address the inefficiency issue of existing general-purpose GPM systems and the non-generality problem of pattern decomposition algorithms. In particular, we propose: 1) a novel partial-embedding programming model, by which users can exploit the considerable performance benefit potential of pattern decomposition easily for general GPM applications (e.g., FSM); 2) an efficient two-level on-the-fly aggregation algorithm that generalizes existing pattern decomposition algorithms to implement the new programming model; 3) two real-world-graph-aware accurate cost models that enable effective searching for optimal pattern decomposition settings; 4) an end-to-end compilation-based system implementation with traditional dataflow optimizing techniques and a new loop rewriting optimization. Our system significantly outperforms state-of-the-art systems (e.g., Automine and Peregrine) by orders of magnitude.

Dec. 2019 | GraphS: Eliminating Redundant Computation and Communication in PIM-Based Graph Processing with Dependence Scheduling (paper appeared in TOCS as an extension version of our SympleGraph PLDI'20 paper, co-first author)

Aug. 2019 | *Advisor: Prof. Xuehai Qian*

We further explore the research opportunities of enforcing precise loop-carried dependency in PIM-based graph processing architectures. To this end, we designed and implemented two PIM-based graph processing architectures (GraphS and its variant GraphSR) supporting dependency propagation. The proposed architectures dramatically over-perform any existing PIM-based systems.

Apr. 2019 | SympleGraph: Distributed Graph Processing with Precise Loop-Carried Dependency Guarantee (paper appeared in PLDI'20, co-first author)

Jul. 2018 | *Advisor: Prof. Xuehai Qian*

We proposed a distributed graph processing system with a precise loop-carried dependency guarantee. Existing distributed graph processing systems neglect the loop-carried dependency expressed in user-defined functions, leading to unnecessary computation and communication. Enforcing loop-carried dependency in distributed graph processing is challenging due to the reduced parallelism and extra communication cost. Correspondingly, we proposed: 1) using circulant scheduling to retain high-performance with reduce parallelism while enabling dependency propagation; 2) propagating dependency selectively to reduce communication overhead; 3) double buffering dependency data to hide communication latency. In the end, our system can outperform state-of-the-art baselines (Gemini and D-Galois) by up to 2.30x and 7.76x.

Social Activities

2016 - 2017 Student Association of Literature, Tsinghua U

2015 - 2016 Student Association of Psychology, Tsinghua U