Bo Zhao | Assistant Professor

Department of Computer Science, **Aalto University**
Konemiehentie 2, 02150 Espoo, Finland
zbjob.github.io • Email: bo.zhao@aalto.fi

Research Interest

I conduct research on efficient ***data-intensive systems*** that translate ***data*** into ***value*** for decision making. The scope of my research spans across multiple subfields, from **scalable reinforcement learning systems** to **distributed data stream management systems**, as well as **code optimization** techniques. That is to answer the question ***"how to co-design multiple layers of the software stack to improve scalability, performance, and energy efficiency of machine learning systems"***. My long-term goal is to explore and understand the fundamental connections between data management and modern machine learning systems to make decision-making more transparent, robust and efficient.

Education

PhD in Computer Science — **Berlin, Germany**
Humboldt-Universität zu Berlin — *02/2016–05/2022*
Thesis: State Management for Efficient Event Pattern Detection
Supervisor: Prof. Dr. Matthias Weidlich
Honors: *magna cum laude*

M.Sc. in Computer Science — **Xi'an, China**
Xi'an Jiaotong University — *09/2012–07/2015*
Thesis: Dependence-Based Coarse-Grained Automatic Parallelisation
Ranking: Top 1% of the university
Honors: *summa cum laude*

B.Sc in Computer Science — **Wuhan, China**
Wuhan Institute of Technology — *09/2008–07/2012*
Thesis: Energy-Aware Routing Optimizations in Wireless Sensor Networks
Ranking: Top 1% of the university
Honors: *summa cum laude*

Research Visits

University of Queensland — **Brisbane, Australia**
Visiting PhD Student in Computer Science, hosted by Prof. Xiaofang Zhou — *05/2017–06/2017*
Topic: Efficient Data Stream Processing

RWTH-AACHEN University — **Aachen, Germany**
Visiting M.Sc. Student in Computer Science, hosted by Prof. Felix Wolf — *09/2013–02/2015*
Topic: High Performance Computing

Work Experience

09/2023-present: Assistant Professor, **Aalto University, Espoo, Finland**
01/2023-08/2023: Assistant Professor (UK Lecturer), **Queen Mary University of London, London, UK**
Honorary Research Fellow, **Imperial College London, London, UK**
07/2021–01/2023: Postdoctoral Researcher, **Imperial College London, London, UK**
02/2016–06/2021: Research Assistant in **Humboldt-Universität zu Berlin, Berlin, Germany**
06/2019–09/2019: Software Development Engineer Intern at Amazon, **AWS Redshift, Berlin, Germany**
11/2015–01/2016: Research Assistant in **Technische Universität Darmstadt, Darmstadt, Germany**
10/2013–02/2015: Student Research Assistant in **RWTH-AACHEN University, Aachen, Germany**

Honors & Awards

2019-2020: Travel Grant of the Silk Road International Symposium for Distinguished Young Scholars
2017–2018: IEEE ICDE Student Travel Grant
2015–2016: EDBT Summer School Travel Grant
2014–2015: Outstanding Graduate, ACM SIGPLAN Travel Grant, ACM SIGMICRO Travel Grant
2012–2013: China National Scholarship(top 0.2%), Creative-Master Scholarship, Excellent Master Student
2011–2012: Excellent Graduation Thesis, Outstanding Graduate

2010–2011: China National Scholarship(top 0.2%), Top Grade Scholarship, Pacemaker to Merit Student, Advanced Individual in Social Practice

2009–2010: China National Scholarship(top 0.2%), Top Grade Scholarship, Pacemaker to Merit Student

2008–2009: Top Grade Scholarship, Pacemaker to Merit Student, Outstanding League Member

Publications

- Huanzhou Zhu*, Bo Zhao*, Gang Chen, Weifeng Chen, Yijie Chen, Liang Shi, Yaodong Yang, Peter Pietzuch, Lei Chen (*equal contribution): **MSRL: Distributed reinforcement learning with dataflow fragments**, *In Proc. of the USENIX Annual Technical Conference (**ATC'23**), Boston, MA, USA, July, 2023*

- Song Liu, Xinhe Wan, Zengyuan Zhang, Bo Zhao, Weiguo Wu: **TurboStencil: You Only Compute Once for Stencil Computation**, ***Future Generation Computer Systems*** *(IF=7.307), 2023*

- Gururaghav Raman, Bo Zhao, Jimmy Chih-Hsien Peng, Matthias Weidlich: **Adaptive incentive-based demand response with distributed non-compliance assessment**, ***Applied Energy*** (IF=11.446), Volume 326, November, 2022

- Bo Zhao : **State Management for Efficient Event Pattern Detection**, *Dissertation*, Humboldt-Universität zu Berlin, Mathematisch-Naturwissenschaftliche Fakultät, May 2022

- Bo Zhao, Han van der Aa, Nguyen Thanh Tam, Nguyen Quoc Viet Hung, Matthias Weidlich: **EIRES: Efficient Integration of Remote Data in Event Stream Processing**, *In Proc. of the 47th ACM SIGMOD International Conference on Management of Data (**SIGMOD'21**), Xi'an, China, ACM, June 2021*

- Bo Zhao, Nguyen Quoc Viet Hung, Matthias Weidlich: **Load Shedding for Complex Event Processing: Input-based and State-based Techniques**, *In Proc. of the 36th IEEE International Conference on Data Engineering (**ICDE'20**), Dallas, TX, USA, IEEE, April 2020*

- Gururaghav Raman, Jimmy Chih-Hsien Peng, Bo Zhao, Matthias Weidlich: **Dynamic Decision Making for Demand Response through Adaptive Event Stream Monitoring**, *In Proc. of the IEEE Power & Energy Society General Meeting (**PESGM'19**), Atlanta, GA, USA. IEEE, August 2019.*

- Bo Zhao: **Complex Event Processing under Constrained Resources by State-based Load Shedding**, *In Proc. of the 34th IEEE International Conference on Data Engineering (**ICDE'18**), Paris, France, IEEE, April 2018*

- Bo Zhao, Zhen Li, Ali Jannesari, Felix Wolf, Weiguo Wu: **Dependence-Based Code Transformation for Coarse-Grained Parallelism**, *In Proc. of the International Workshop on Code Optimisation for Multi and Many Cores (**COSMIC'15**) held in conjunction with **CGO'15**, San Francisco Bay Area, CA, USA, ACM, February 2015*

- Bo Zhao, Ali Jannesari: **Dependence-Based Parallel Code Generation Using Intel CnC**, *In Proc. of the 24th International Conference on Parallel Architectures and Compilation Techniques (**PACT'15**),San Francisco Bay Area, CA, USA, October 2015* (ACM SRC poster)

- Zhen Li, Bo Zhao, Ali Jannesari, Felix Wolf: **Beyond Data Parallelism: Identifying Parallel Tasks in Sequential Programs**, *In Proc. of the 15th International Conference on Algorithms and Architectures for Parallel Processing (**ICA3PP'15**), Springer, November 2015*

- Song Liu, Bo Zhao, Qing Jiang, Weiguo Wu: **A Semi-Automatic Coarse-Grained Parallelization Approach for Loop Optimization And Irregular Code Sections** (in Chinese). *Chinese Journal of Computers, 2017*

- Song Liu, Weiguo Wu, Bo Zhao, Qing Jiang: **Loop Tiling for Optimization of Locality and Parallelism** (in Chinese). *Journal of Computer Research and Development, 2015*

Research Talks

June 2023: Invited guest lecture at tHumboldt-Universität zu Berlin, Berlin, Germany;

June 2023: Invited talk at the Huawei Cloud InnovWave Overseas Workshop, Munich, Germany;

June 2023: Invited guest lecture at TU Wien, Vienna, Austria;

May 2023: Invited talk at the Global Software Technology Summit, Dresden, Germany;

May 2023: Invited talk at the Max Planck Institute for Software Systems (MPI-SWS), Saarbrücken, Germany;

March 2023: Invited talk at Aalto University, Espoo, Finland;

January 2023: Invited talk at TU Wien, Vienna, Austria;
November 2022: Invited talk at King's College London, London, UK;
December 2021: Invited talk at Xi'an Jiaotong University, Virtual Event, China;
November 2021: Invited talk at Nanjing University, Virtual Event, China;
June 2021: The 47th ACM International Conference on Management of Data (*SIGMOD'21*), Virtual Event, China;
March 2021: Invited talk at EPFL, Lausanne, Switzerland;
December 2020: Invited talk at Hasso Plattner Institute, Potsdam, Germany;
November 2020: Invited talk at ETH Zürich, Zürich, Switzerland;
November 2020: Invited talk at Imperial College London, London, UK;
November 2020: Invited talk at Technical University of Berlin, Berlin Germany;
April 2020: The 36th IEEE International Conference on Data Engineering (*ICDE'20*), Dallas, TX, USA;
April 2019: Invited talk at Xi'an Jiaotong University, Xi'an, China;
April 2018: The 34th IEEE International Conference on Data Engineering (*ICDE'18*), Paris, France;
September 2015: The 7th Annual Concurrent Collections Workshop (*with LCPC'15*), Raleigh, NC, USA;
September 2015: The 44th International Conference on Parallel Processing (*ICPP'15*), Beijing, China;
February 2015: The 2nd International Workshop on Code Optimisation for Multi and Many Cores (*COSMIC'15*), San Francisco Bay Area, CA, USA;
September 2014: The Sixth Annual Concurrent Collections Workshop, Intel Corp in Hillsboro, OR, USA;

Academic Services

Program Committees: The Conference on Information and Knowledge Management (***CIKM***) 2021, 2022, 2023
Availability Committees: The ACM International Conference on Management of Data (***SIGMOD***) 2022
Demonstrations Track Program Committees: IEEE International Conference on Data Engineering (***ICDE***) 2023
Reviewers for Journals: IEEE Transactions on Parallel and Distributed Systems (***TPDS***) 2023, Journal of Systems and Software (***JSS***) 2016

Teaching Experience

Semester B 2023: ECS656U *Distributed Systems*, Queen Mary University of London
Summer semester 2020: Seminar on *Distributed Data Management Systems*, Humboldt-Universität zu Berlin
Summer semester 2020: Oral exam examiner on *Process Mining*, Humboldt-Universität zu Berlin
Winter semester 2019: Oral exam examiner on *Event Process*, Humboldt-Universität zu Berlin
Winter semester 2019: Exercises (Übung) on *Data Stream Processing*, Humboldt-Universität zu Berlin
Summer semester 2018: Oral exam examiner on *Process Mining*, Humboldt-Universität zu Berlin
Summer semester 2018: Seminar on *Event Stream Processing*, Humboldt-Universität zu Berlin

Student Mentoring

- Master thesis project on "Dataflow-Based MLOps for Machine Learning Pipelines", Mr. Vishnu Puramchalil, December 2022-present, Queen Mary University of London, UK
- Master thesis project on "Adaptive Query Analytics over Dynamic Data Streams", Mr. Shaurya Rana, December 2022-present, Queen Mary University of London, UK
- Master thesis project on "Dataflow Optimisation for Scalable Reinforcement Learning Systems", Mr. Mustapha Abdullahi, December 2022-present, Queen Mary University of London, UK
- Master thesis project on "Distributed Data Stream Processing for Business Intelligence", Mr. Chinar Amrutkar, December 2022-present, Queen Mary University of London, UK
- Student intern project on "Implementing MuZero Agents Using Mindspore Computation Graphs", Mr. Liyi Tan, June 2022-present, Imperial College London, UK
- Undergraduate project on "Implementing MuZero Algorithm Using the Mindspore DL Engine", Mr. Bartłomiej Cieślar, January 2022-May 2022, Imperial College London, UK
- Master thesis project on "Mining Constraints to Optimise CEP Load Shedding for Multiple Queries", Mr. Xudong Zhu, 2019-2020, Humboldt-Universität zu Berlin, Germany

Projects

CloudButton: a Serverless Data Analytics Platform
Funding agency: EU Horizon 2020 Framework Programme
Duration: 2019-2022, Role: Participant (Postdoctoral researcher)
Amount: 4.2 million EUR

Process-Awareness of Event-Driven Systems: Model, Analysis and Optimisation
Funding agency: German Research Foundation (Deutsche Forschungsgemeinschaft, DFG)
Duration: 2014-2021, Role: Participant (Research Assistant)
Amount: 1 million EUR

Detailed Projects

Distributed Reinforcement Learning Systems with Dataflow Fragments **Imperial College London**
07/2021–present

Problem. Reinforcement learning (RL) needs to train a large number of agents, which is resource-intensive and must scale to large GPU clusters. Yet, current distributed RL systems hardcode a single strategy to parallelize and distribute an RL algorithm based on its algorithmic structure and only permit the acceleration of specific parts of the computation (e.g. policy network updates) on GPU workers. Fundamentally, existing systems lack abstractions to decouple RL algorithms from their execution.

Approach. We introduce a new abstraction, *fragmented dataflow graphs*, which offer flexibility in how RL training is parallelized and distributed. A fragmented dataflow graph maps Python functions from an RL algorithm's training loop to parallel computational *fragments*. Fragments can be executed on different devices by translating them to low-level intermediate dataflow representations, e.g. computational graphs supported by deep learning engines (PyTorch, TensorFlow, MindSpore), CUDA implementations or multi-threaded CPU processes. A *distribution policy* governs how fragments are mapped to cluster resources, without requiring changes of the algorithm implementation. Our system subsumes the distribution strategies of existing systems, while scaling RL training to many GPU workers (64 gpus on Azure Cloud.) .

Output. We built our system and reported the evaluation results in a research paper published in ATC'23. In addition, the work has been integrated into MindSpore, a leading industry ML framework, under the name of *MindSpore Reinforcement*. More publications will come soon. Stay tuned.😎

State Management for Efficient Event Pattern Detection **Humboldt-Universität zu Berlin**
02/2016–05/2021

Problem. Complex event processing (CEP) evaluates queries over streams of events for low-latency detection of user-specified patterns which correlate event data within certain time windows. Such queries are stateful and therefore, the CEP engine needs to maintain a set of partial results. When combined with kleene closure operators, the size of partial results grows exponentially in the number of processed events. High input rates of streams amplify this issue. This makes low-latency data analysis challenging. What's worse, when integrating with remote data sources, CEP engines need to fetch them and thus, CEP's performance deteriorates even further by remote data transmission latency.

Approach. I propose strategies for optimized state management in event pattern detection. First, I enable best-effort query evaluation with *load shedding* that discards both input events and partial matches. I carefully select the partial matches and input events to drop in order to satisfy a latency bound while striving for a minimal loss in result quality. Second, to efficiently integrate remote data, I decouple the fetching of remote data from its use in query evaluation through a *caching* mechanism. Based thereon, we hide the transmission latency of remote data by *prefetching* data based on anticipated use and by *lazy evaluation* that postpones the event selection based on remote data to avoid interruptions. A cost model is proposed to determine when to fetch which remote data items and how long to keep them in the cache.

Output. I built a prototype CEP engine in C++ around 7k lines of code from scratch. The work of load shedding has been published in ICDE'18 PhD Symposium and ICDE'20 (source code `https://github.com/zbjob/AthenaCEP`). The work of remote data integration has been published in SIGMOD'21 (source code `https://github.com/zbjob/EIRES`). In addition, I conducted inter-discipline collaboration with the Department of Electronic Engineering, National University of Singapore, to apply our complex event processing optimisations to smart grid management up to 1.6 million residents (source code `https://github.com/zbjob/SmartGrid`). The results has been published in IEEE PES-GM'19. More comprehensive evaluations are presented in a journal article in Applied Energy'22.

Efficient Profiling for Cloud-Based Data Warehouse **Amazon Web Services, Redshift Team**
06/2019–09/2019

Problem. Redshift is a leading cloud-based distributed data warehouse deployed on AWS. In order to target the performance bottleneck and pave the way for further improvements, it is necessary to trace and analyse detailed query execution behaviours at runtime. However, profiling at extremely fine granularity incurs unacceptably high computational overhead. My goal is to bridge this gap.

Approach. I built the performance analysis tool based on eBPF (extended Berkeley Packet Filter) and its front end BCC (BPF Compiler Collection). In order to reduce the computational overhead, the eBPF code monitors statistics in kernel space and stores them in BPF maps. In user space, I fetch the BPF maps via BCC APIs (through Python scripts) and perform sophisticated analysis asynchronously. To further reduce the computational overhead, I employ the approximation technique (e.g. sketching) and sampling techniques to monitor the execution time for different code sections. I integrate this profiler into the code generator of the query plan. Therefore, the probes have been automatically inserted into the generated C++ code that is compiled from PostgreSQL queries. When the queries are executed, the profiling is automatically done.

Output. I evaluated the performance analysis tool against the TPC-DS benchmark at 3TB and 10TB scale on clusters of AWS DC2.8xlarge instances. The tool is able to reduce the computational overhead to 1.0% and obtain accurate profiling information. I also discovered insights for performance improvements on real-world business workloads in Redshift on AWS. For instance, one query of a Redshift customer has been improved by 1.75 $\times$ faster.

Dependence-Based Auto-Parallelisation on Multicore systems — RWTH-Aachen University

10/2013–02/2015

Problem. Many software products implemented sequentially have failed to exploit the potential parallelism of multicore architectures. Significant re-engineering and refactoring of existing software is needed to support the use of new hardware features. Due to the high cost of manual transformation, an automated approach to transforming existing software and taking advantage of multicore architectures is highly beneficial.

Approach. I proposed and developed a novel auto-parallelisation framework, which integrates data-dependence profiling, coarse-grained task parallelism extraction and source-to-source code transformation. To this end, I identified code sections called computational units (CUs), which follow a read-compute-write pattern and can be used as building blocks to construct parallel tasks. Specifically, I first used an in-house profiling tool (based on LLVM) to analyse the data and control dependence among the source code and therefore, generated the dependence graph of CUs. Then I merged CUs in the CU graph and generated a more coarse-grained task graph. Second, I used LLVM front-end Clang to parse the source code, extract the AST, traverse the AST to locate DOALL loops and the code sections targeted by the task graph. Finally, I transformed the sequential source code to parallel one using Intel Threading Building Blocks (TBB) and Intel Concurrent Collections (CnC).

Output. My work has been published in two conference papers, COSMIC@CGO'15 and ICA3PP'15. In addition, it has been integrated into the project of "Discovery of Potential Parallelism" (DiscoPoP) (`https://www.discopop.tu-darmstadt.de`).

Cache-Efficient Loop Optimisation on NUMA Systems — Xi'an Jiaotong University

09/2012–06/2015

Problem. On modern many-core CPUs and NUMA systems, it is necessary to exploit the complex memory subsystems and the multi-level cache hierarchy for parallelism, high performance and therefore, energy efficiency.

Approach. I focused on data-intensive parallel loop optimisations to exploit pipeline/wave-front parallelism by loop transformations. To this end, I proposed a loop tiling strategy for imperfectly-nested loop nests and leveraged machine learning technologies to select the appropriate tile size to keep the cache hot.

Output. I successfully exploited fine-grained parallelism and data locality, achieving speedups up to 196$\times$. The results have been published in two journal articles.

References (with website links on names)

Name	Affiliation	Email address
Prof. Peter Pietzuch	Imperial College London, UK	prp@imperial.ac.uk
Prof. Matthias Weidlich	Humboldt-Universität zu Berlin, Germany	matthias.weidlich@hu-berlin.de
Dr. Nguyen Quoc Viet Hung	Griffith University, Australia	henry.nguyen@griffith.edu.au
Prof. Han van der Aa	Universität Mannheim, Germany	han@informatik.uni-mannheim.de