Louis Jenkins

Ardmore, PA 19003
LouisJenkinsCS@hotmail.com • +1 (610) 931-1207
https://www.linkedin.com/in/LouisJenkinsCS • http://github.com/LouisJenkinsCS • http://LouisJenkinsCS.github.io

EDUCATION

UNIVERSITY OF ROCHESTER , Rochester, NY *Fall 2018 – Summer 2023*

- **Degree:** Masters of Science (M.S.) in Computer Science
- **Fellowships:**
  - Provost Fellowship *Fall 2018 - Summer 2019*
  - Department of Energy Computational Science Graduate Fellowship *Fall 2019 – Summer 2023*

BLOOMSBURG UNIVERSITY OF PENNSYLVANIA , Bloomsburg, PA *Summer 2012 - Fall 2017*

- **Degree:** Bachelor of Science (B.S.) in Computer Science
- **Scholarship:** Board of Governors *Fall 2013 - Spring 2017*

RESEARCH EXPERIENCE

STUDENT RESEARCHER , Lehigh University, Bethlehem, PA *Summer 2016*

- **Project:** Concurrent and Scalable Built-in Hash Table for the Go Programming Language [1]
- **Advisor:** Michael F. Spear
- **Grant:** National Science Foundation
- **Awards:**
  - Peer's Choice for Outstanding Project.
  - Honorable Mention for CRA 2017 Outstanding Undergraduate Researchers, sponsored by Microsoft Research.
- **Summary:**
  - Designed and implemented scalable lock-based concurrent map for Golang, modifying the runtime and compiler.
  - Implemented with compatibility for Go map syntax; supports insert/lookup/remove and concurrent iteration.
  - Outperforms sequential map by up to 7x across diverse microbenchmarks, competitive against lock-free maps.

GOOGLE SUMMER OF CODE (STUDENT) *Summer 2017*

- **Project:** Distributed Data Structures [2]
- **Advisors:** Michael Ferguson, Engin Kayraklioglu
- **Sponsor:** Google Summer of Code
- **Summary:**
  - Designed and implemented the first scalable ordered data structure for PGAS languages (100x @ 3072 Processors).
  - Designed and implemented a novel scalable unordered data structure (500x @ 3072 Processors).
  - Key role in designing Collections modules; all officially available in Chapel v1.16.

STUDENT RESEARCHER , Bloomsburg University, Bloomsburg, PA *Fall 2017*

- **Project:** Introducing LLVM to the Java Virtual Machine
- **Advisor:** William Calhoun
- **Grant:** Professional Experience Grant
- **Summary:**
  - Engineered a prototype frontend capable of converting JVM Classfiles into LLVM Modules.
  - Explored the feasibility of using LLVM as a backend and optimizer for a Just-In-Time (JIT) Compiler.
  - Conceptualized and developed a proof-of-concept that successfully executes basic Java programs.

RESEARCH ASSISTANT , University of Rochester, Rochester, NY *Fall 2018 – Fall 2021*

- **Project** Persistent Memory Analysis Tool (PMAT)
- **Advisor** Michael Scott
- **Sponsor:** Department of Energy Computational Science Graduate Fellowship
- **Summary:**
  - Developed a persistent memory consistency-checker utilizing just-in-time instrumentation and shadow memory.
  - Conducted simulations of Out-of-Order Write-Back to illustrate issues with volatile CPU cache and non-volatile memory.
  - Demonstrated proficiency in identifying both apparent and subtle bugs in synchronized programs.

RESEARCH ASSISTANT , University of Rochester, Rochester, NY *Spring 2022 – Summer 2023*

- **Project:** Dynamic Resource Scheduling for Jupyter Notebooks
- **Advisor:** Sreepathi Pai
- **Sponsor:** Department of Energy Computational Science Graduate Fellowship
- **Summary:**
  - Designed a Resource Scheduler capable of scheduling at the granularity of individual Jupyter Notebook cells.
  - Optimized computational resource utilization by leveraging periods of interactivity and idle time.
  - Employed Offline Traces to estimate resource usage of individual cells, enabling informed scheduling decisions.

WORK EXPERIENCE

PACIFIC NORTHWEST NATIONAL LABORATORY

- **Position:**
  - Technical Intern IV — Summer 2018
  - PhD Intern (Part Time) — Fall 2018 - Spring 2019
  - PhD Intern (Full Time) — Summer 2019
  - Non-Employee Volunteer — Fall 2019 – Summer 2020
- **Awards:** Outstanding Performance Award
- **Project:** Chapel HyperGraph Library (CHGL) [3,4,5]
  - Led the development of a distributed global-view hypergraph library using Chapel programming language [3].
  - Created several libraries and abstractions specifically designed for handling irregular patterns, including aggregation [4].
  - Collaborated with HyperNetX (HNX) to explore the modeling of DNS data and relationships as hypergraphs [5,9].

MENTORING EXPERIENCE

RAILS GIRLS SUMMER OF CODE (COACH) , *Summer 2018*

- **Project:** Distributed Sorting Algorithms in Chapel
- **Summary:**
  - Provided guidance and mentorship in design, implementation, and profiling bottlenecks.
  - Identified and addressed significant performance issues in the language during the mentorship.
  - Resulted in a bootstrap for further development of distributed radix sort in Chapel.

GOOGLE SUMMER OF CODE (MENTOR) *Summer 2019*

- **Project:** Concurrent-Safe Memory Reclamation System
- **Summary:**
  - Co-designed and assisted in implementation of epoch-based reclamation in shared- and distributed-memory [8].
  - Epoch-Based Manager scaled in distributed memory on Cray-XC; tested up to 64 nodes
  - Contributed Lock Free Queue & Stack, Scalable Disributed Atomics, and Epoch-Based Manager.

TEACHING ASSISTANT

UNIVERSITY OF ROCHESTER

- **Course #1:** Parallel and Distributed Computing — *Spring 2019*
- **Course #2:** Programming Language Design, and Implementation — *Fall 2020*
- **Award:** Outstanding Graduate Teaching Assistant Award — *Spring 2020*
- **Summary:**
  - Supervise undergraduate teaching assistants and manage task allocation.
  - Contribute to the design and creation of homework assignments and automated grading systems.
  - Conduct regular office hours and foster a supportive learning atmosphere for students.

SKILLS

PROGRAMMING LANGUAGES

- **Programming Paradigms:**
  - **Fluent:** Object-Oriented (OOP), Scripting, Imperative, Generic, Concurrent
  - **Proficient:** Declarative
  - **Intermediate:** Functional, Reactive, Array-oriented
  - **Beginner:** Logic
- **Programming Languages:**
  - **Fluent:** C, C++, Python, Java, Chapel, Bash
  - **Proficient:** x86 Assembly
  - **Intermediate:** Haskell, R, Prolog, SQL
  - **Beginner:** Rust, Julia, C, Kotlin, Scala

TECHNOLOGIES

- **Technologies:**
  - **Proficient:** OpenMP, MPI, GASNet, Compilers, Debugging, High-Performance Computing (HPC), Jupyter Notebooks
  - **Intermediate:** Linux System Administration (Debian, Fedora, Arch), Version Control (Git), Memory Management
  - **Beginner:** Cloud Computing (AWS EC2), Agile Development (SCRUM, JIRA), Database Administration (Sqlite3)

PERSONAL PROJECTS

MOLTAROS - HOBBY OPERATING SYSTEM

- **GitHub Repository:** LouisJenkinsCS/MoltarOS
- **Summary:**
  - Developed an Operating System, Kernel, and Bootloader using C and Assembly, guided by OSDev Tutorials.
  - Designed the system to manage basic interrupts, low-level drivers, memory management, and rudimentary multitasking.
  - Ensured the system's capability to boot in a virtual machine and facilitate basic interaction with a human operator.

CHAPEL-ATOMIC-OBJECTS - NON-BLOCKING PRIMITIVES FOR PGAS

- **GitHub Repository:** LouisJenkinsCS/Chapel-Atomic-Objects
- **Summary:**
  - Designed concurrent-safe resizable distributed arrays which were 40x faster to resize than language built-in. [2]
  - Experimented with Software Transactional Memory for PGAS using non-blocking high-level primitives.
  - Implemented Remote-Memory Access (RMA)-enabled scalable atomic operations on arbitrary distributed objects.

QUIESCENT STATE-BASED RECLAMATION (QSBR) FOR PGAS

- **GitHub Repository:** chapel-lang/chapel-attic/tree/collaborators/QSBR
- **Changelog:** chapel-lang/chapel-attic/commit/acfe31af56974edc5f1c0ea946ef79adda52005d
- **Summary:**
  - Designed memory management in Chapel based on QSBR that enabled users to delete objects with concurrent-safety.
  - Implementation provided negligible overhead and maintained consensus in distributed contexts.
  - Integrated into Chapel core but promptly reverted due to compatibility issues across other systems.

SWISS-ARMY-KNIFE (S.A.K) OVERLAY

- **GitHub Repository:** LouisJenkinsCS/S.A.K-Overlay
- **Summary:**
  - Designed Android application which enabled snapping and movement of pre-made or user-made widgets.
  - Persisted state of widgets (position, contents, history, etc.) via serialization to and from sqlite3 database.
  - Prototyped a Drag-and-Drop WYSIWYG editor for user-made widgets and JIT compilation of user-defined code.
  - Used Java 8 features (backports) like Streams, Observables, event buses, and other reactive-programming components.

PUBLICATIONS

[1] L. Jenkins, T. Zhou, and M. Spear, "Redesigning Go‘s Built-In Map to Support Concurrent Operations," in 2017 26th International Conference on Parallel Architectures and Compilation Techniques (PACT), 2017, pp. 14–26.

[2] L. Jenkins, "RCUArray: An RCU-Like Parallel-Safe Distributed Resizable Array," in 2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), 2018, pp. 925–933.

[3] L. Jenkins, et al., "Chapel HyperGraph Library (CHGL)," 2018 IEEE High Performance extreme Computing Conference (HPEC), pp. 1–6, 2018.

[4] L. Jenkins, M. Zalewski, and M. Ferguson, "Chapel Aggregation Library (CAL)," in 2018 IEEE/ACM Parallel Applications Workshop, Alternatives To MPI (PAW-ATM), 2018, pp. 34–43.

[5] C. Joslyn, ..., and L. Jenkins, et al., ”Hypergraph analytics of domain name system relationships," in 17th International Workshop on Algorithms and Models for the Web-Graph (WAW 2020). pp. 1-15.

[6] L. Jenkins and M. Zalewski, "Chapel Graph Library (CGL)," in Proceedings of the ACM SIGPLAN 6th on Chapel Implementers and Users Workshop, New York, NY, USA, 2019, pp. 29–30.

[7] L. Jenkins, J. S. Firoz, M. Zalewski, C. Joslyn and M. Raugas, ”Graph Algorithms in PGAS: Chapel and UPC++," 2019 IEEE High Performance Extreme Computing Conference (HPEC), Waltham, MA, USA, 2019, pp. 1-6.

[8] G. Dewan, and L.Jenkins, ”Paving the way for Distributed Non-Blocking Algorithms and Data Structures in the Partitioned Global Address Space," 2020 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), pp. 659-666

[9] J. S. Firoz, L. Jenkins, et al., ”Computing Hypergraph Homology in Chapel," 2020 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), pp. 667-670