The Journal of Open Source Software

Caffeine: A parallel runtime library for supporting modern Fortran compilers

Dan Bonachea [1*], **Katherine Rasmussen** [1*], **Brad Richardson** [1*], and **Damian Rouson** [1*]

1 Lawrence Berkeley National Laboratory, United States * These authors contributed equally.

DOI: 10.21105/joss.07895

Software

- Review
- Repository
- Archive

Editor: Daniel S. Katz

Reviewers:

- @tclune
- @MichaelSiehl

Submitted: 26 February 2025
Published: 29 March 2025

Summary

The Fortran programming language standard added features supporting single-program, multiple-data (SPMD) parallel programming and loop parallelism beginning with Fortran 2008 (Fortran Standards Committee JTC1/SC22/WG5, Oct 2010). In Fortran, SPMD programming involves the creation of a fixed number of images (instances) of a program that execute asynchronously in shared or distributed memory, except where a program uses specific synchronization mechanisms. Fortran's "coarray" distributed data structures offer a subscripted, multidimensional array notation defining a partitioned global address space (PGAS). One image can use this notation for one-sided access to another image's slice of a coarray.

The CoArray Fortran Framework of Efficient Interfaces to Network Environments (Caffeine) provides a runtime library that supports Fortran's SPMD features (Bonachea et al., 2025; Rouson & Bonachea, 2022). Caffeine implements inter-process communication by building atop the GASNet-EX exascale networking middleware library (Bonachea & Hargrove, 2018, 2024). Caffeine is the first implementation of the compiler- and runtime-agnostic Parallel Runtime Interface for Fortran (PRIF) specification (Bonachea et al., 2024a, 2024b). Any compiler that targets PRIF can use any runtime that supports PRIF. Caffeine supports researching the novel approach of writing most of a compiler's parallel runtime library in the language being compiled: Caffeine is primarily implemented using Fortran's non-parallel features, with a thin C-language layer that invokes the external GASNet-EX communication library. Exploring this approach in open source lowers a barrier to contributions from the compiler's users: Fortran programmers. Caffeine also facilitates research such as investigating various optimization opportunities that exploit specific hardware such as shared memory or specific interconnects.

Figure 1 depicts a software stack in which a parallel runtime library, such as Caffeine, supports compiled Fortran code by implementing PRIF.

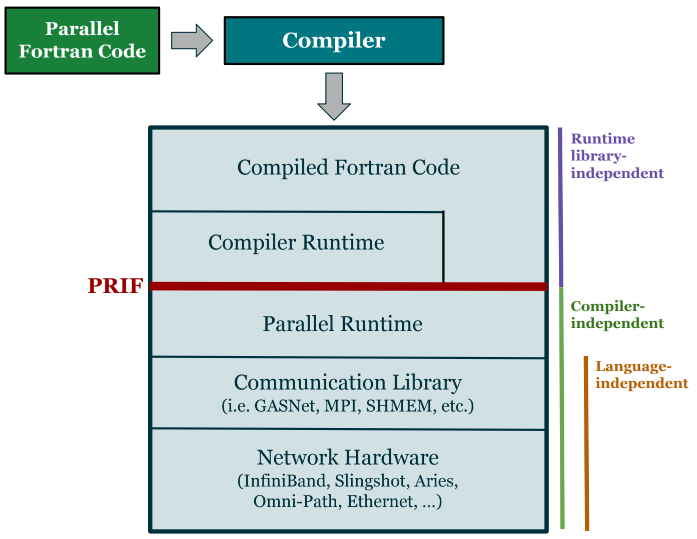
Parallel Fortran Code
Compiler
Compiled Fortran Code
Compiler Runtime
PRIF
Parallel Runtime
Communication Library (i.e. GASNet, MPI, SHMEM, etc.)
Network Hardware (InfiniBand, Slingshot, Aries, Omni-Path, Ethernet, ...)
Runtime library-independent
Compiler-independent
Language-independent

Figure 1: The parallel Fortran software stack enabled by the Caffeine parallel runtime's implementation of PRIF, reproduced with permission from (Bonachea et al., 2024a).

Statement of need

The Coarray Fortran domain-specific language pioneered a parallel programming approach designed as a syntactically small extension to Fortran 95 (Numrich & Reid, 1998). Fortran 2008 incorporated Coarray Fortran features, including multi-image execution, synchronization statements, coarrays, and more. Fortran 2018 greatly expanded this feature set to include such concepts as teams (groupings) of images, events (counting semaphores), collective subroutines and failed-image detection (fault tolerance). Fortran 2023 provided additional, minor multi-image extensions, including notified remote data access (Fortran Standards Committee JTC1/SC22/WG5, Nov 2023).

Caffeine's initial target compilers include LLVM `flang` and LFortran, both of which have no existing multi-image parallel runtime and thus will need one to reach full compliance with the 2008, 2018, or 2023 versions of the Fortran standard. The Caffeine project team has submitted the PRIF specification as a pull request on the `llvm-project` `git` repository and through private correspondence have confirmed the lead LFortran developer's interest in adopting PRIF when LFortran begins work on enabling multi-image execution.

State of the field

At least eight implementations of Fortran's multi-image features have been developed:

1. Caffeine,
2. g95 (Vaught, 2013),
3. HPE Cray Compiling Environment (CCE) Fortran compiler (Hewlett Packard Enterprise, 2025),
4. Intel Fortran Compiler (Intel Corporation, 2025),
5. Numerical Algorithms Group (nAG) Fortran compiler (Numerical Algorithms Group Ltd, 2025),
6. OpenCoarrays and the GNU Compiler Collection (GCC) (Fanfarillo et al., 2014),
7. Rice University Coarray Fortran compiler (Dotsenko et al., 2004), and
8. University of Houston OpenUH compiler (Chapman et al., 2013; Ge, 2016).

One can view these along several axes: open- to closed-source, portable to hardware-specific, research artifact to production-ready, and dormant to actively developed. One can also categorize each runtime in terms of the choice of communication substrate and the ability to switch substrates. In each measure, Caffeine stands apart from some or all the other compilers.

Activity level

Of the eight aforementioned parallel runtimes, three appear to be dormant: g95, Rice University, and the University of Houston. The rest remain under active development.

Openness and portability

As commercial products with proprietary, hardware-specific optimizations, the HPE, Intel, and nAG compilers all have closed-source runtime libraries. Caffeine differs from such runtimes in its open-source development practices. The openness of a project's source impacts other dimensions of comparison. For example, openness impacts portability: compilers and runtimes delivered as precompiled binary files have limited portability across hardware architectures.

In addition to portability across hardware, one can compare a parallel runtime library's portability across compilers. With the exception of Caffeine, each of the eight aforementioned runtimes is designed to support only one compiler. In contrast, any compiler that targets PRIF can use the Caffeine runtime library.

Communication substrates

One of Caffeine's most unique traits lies in its use of the GASNet-EX networking middleware library. Developed as part of the Exascale Computing Project funded by the United States Department of Energy (DOE), GASNet-EX facilitates communication for PGAS programming models on supercomputers at DOE leadership computing facilities. GASNet-EX often outperforms the widely used Message Passing Interface (MPI) communication library; Figure 2 compares communication performance on the Frontier supercomputer that ranked first from November 2022 through November 2024 on the Top 500 list of the world's fastest general-purpose supercomputers.

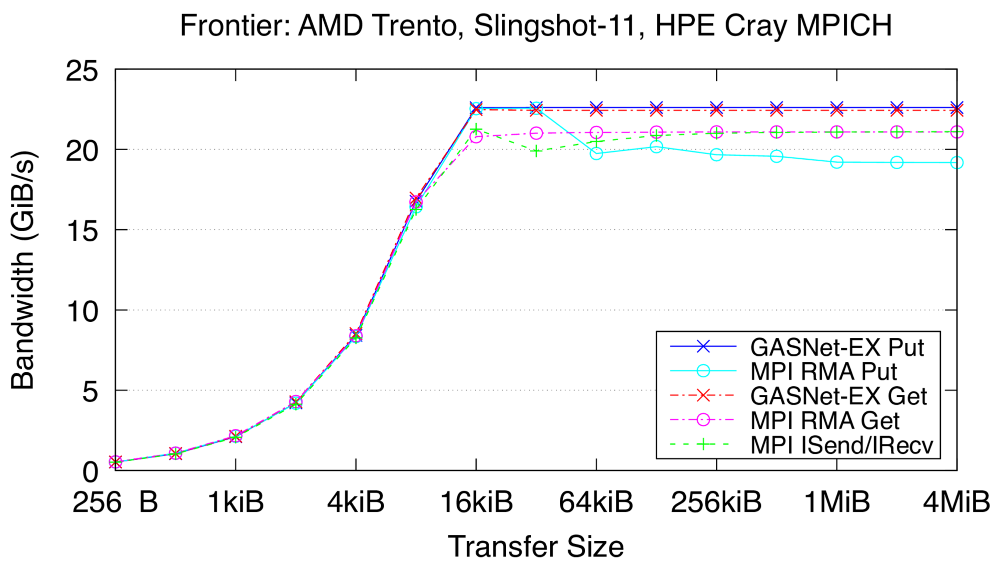

Figure 2: Comparison of GASNet-EX and MPI bandwidth on the Frontier supercomputer, reproduced with permission from (Lawrence Berkeley National Laboratory, Apr 2023).

The University of Houston and Rice University compilers both implement inter-process communication by calling the GASNet-1 library (Bonachea, 2002), a predecessor to the modern GASNet-EX library used by Caffeine.

By contrast, the GCC Fortran compiler (`gfortran`) supports multi-image execution by linking programs to the OpenCoarrays runtime library, which in turn uses MPI for inter-process communication. Similarly, the Intel compiler uses MPI in the version compiled for execution in distributed-memory. A different version of the Intel compiler produces executable programs that work only in shared memory, but details of the corresponding communication substrate are not public.

HPE's Cray Compilation Environment (CCE) Fortran compiler uses a proprietary closed-source runtime. The CCE Fortran runtime has historically targeted HPE's proprietary DMAPP communication library.

Research suitability

Open-source development expands the ways in which software can support research. For example, the ability to recompile a runtime library facilitates studying performance portability or studying the impact of different build configurations. Although the University of Houston and Rice University compilers are research compilers, both appear to be dormant (Chapman et al., 2013; Mellor-Crummy, 2011). OpenCoarrays and Caffeine are the only two actively maintained open-source runtimes and thus the two most suitable for research.

Recent research and scholarly publications

Caffeine has supported researching whether and how one can write a compiler's parallel runtime library in the language being compiled. This research spawned a peer-reviewed publication describing the approach developed for Caffeine (Rouson & Bonachea, 2022). Caffeine also inspired and supported researching whether and how one can define an interface to a compiler's parallel runtime in a compiler- and runtime-agnostic manner. The latter research produced a second peer-reviewed publication motivating and describing PRIF (Bonachea et al., 2024a). Future research will include investigating avenues for supporting the optimization of communication on specific categories of hardware.

Acknowledgments

The Computer Languages and Systems Software (CLaSS) Group at Berkeley Lab (Lawrence Berkeley National Laboratory, 2025) has developed Caffeine on funding from the Exascale Computing Project (ECP) and the Stewardship for Programming Systems and Tools (S4PST) project.

References

Bonachea, D. (2002). *GASNet specification, v1.1* (UCB/CSD-02-1207). University of California, Berkeley. https://doi.org/10.25344/S4MW28

Bonachea, D., & Hargrove, P. H. (2018). GASNet-EX: A high-performance, portable communication library for exascale. *Proceedings of Languages and Compilers for Parallel Computing (LCPC'18), 11882*. https://doi.org/10.25344/S4QP4W

Bonachea, D., & Hargrove, P. H. (2024). *GASNet specification collection, revision 2024.5.0* (LBNL-2001595). Lawrence Berkeley National Laboratory. https://doi.org/10.25344/S4160B

Bonachea, D., Rasmussen, K., Richardson, B., & Rouson, D. (2024a). Parallel runtime interface for Fortran (PRIF): A multi-image solution for LLVM Flang. *Tenth Workshop on the LLVM Compiler Infrastructure in HPC (LLVM-HPC2024)*. https://doi.org/10.25344/S4N017

Bonachea, D., Rasmussen, K., Richardson, B., & Rouson, D. (2024b). *Parallel runtime interface for Fortran (PRIF) specification (rev. 0.5)*. Lawrence Berkeley National Laboratory (LBNL), Berkeley, CA (United States). https://doi.org/10.25344/S4CG6G

Bonachea, D., Rasmussen, K., Richardson, B., & Rouson, D. (2025). Caffeine: CoArray Fortran framework of efficient interfaces to network environments. In *GitHub repository*. GitHub. https://github.com/berkeleylab/caffeine

Chapman, B., Eachempati, D., & Hernandez, O. (2013). Experiences developing the OpenUH compiler and runtime infrastructure. *International Journal of Parallel Programming*, *41*, 825–854. https://doi.org/10.1007/s10766-012-0230-9

Dotsenko, Y., Coarfa, C., & Mellor-Crummey, J. (2004). A multi-platform co-array Fortran compiler. *Proc. 13th International Conference on Parallel Architecture and Compilation Techniques (PACT)*. https://doi.org/10.1109/PACT.2004.1342539

Fanfarillo, A., Burnus, T., Cardellini, V., Filippone, S., Nagle, D., & Rouson, D. (2014). OpenCoarrays: Open-source transport layers supporting coarray Fortran compilers. *Partitioned Global Address Space Programming Models (PGAS)*. https://doi.org/10.1145/2676870.2676876

Fortran Standards Committee JTC1/SC22/WG5. (Nov 2023). *Information technology – Programming languages – Fortran, ISO/IEC 1539-1:2023*. International Organization for Standardization (ISO).

Fortran Standards Committee JTC1/SC22/WG5. (Oct 2010). *Information technology – Programming languages – Fortran, ISO/IEC 1539-1:2010*. International Organization for Standardization (ISO).

Ge, S. (2016). *Implementation and evaluation of additional parallel features of Coarray Fortran* [Master's thesis, University of Houston]. http://hdl.handle.net/10657/3268

Hewlett Packard Enterprise. (2025). Cray Compiling Environment Fortran compiler. In *web page*. https://cpe.ext.hpe.com/docs/latest/getting_started/CPE-CCE-Fortran.html

Intel Corporation. (2025). Intel Fortran Compiler. In *web page*. https://www.intel.com/content/www/us/en/developer/tools/oneapi/fortran-compiler.html

Lawrence Berkeley National Laboratory. (2025). Computer Languages & Systems Software Group. In *Web page*. https://go.lbl.gov/class

Lawrence Berkeley National Laboratory. (Apr 2023). GASNet-EX 2023 performance examples. In *Web page*. https://gasnet.lbl.gov/performance-2023

Mellor-Crummy, J. (2011). Coarray Fortran 2.0. In *web page*. http://caf.rice.edu

Numerical Algorithms Group Ltd. (2025). nAG Fortran compiler. In *web page*. https://nag.com/fortran-compiler/

Numrich, R. W., & Reid, J. K. (1998). Co-array Fortran for parallel programming. *ACM Fortran Forum*, *17*(2), 1–31. https://doi.org/10.1145/289918.289920

Rouson, D., & Bonachea, D. (2022). Caffeine: CoArray Fortran framework of efficient interfaces to network environments. *2022 IEEE/ACM Eighth Workshop on the LLVM Compiler Infrastructure in HPC (LLVM-HPC)*, 34–42. https://doi.org/10.25344/S4459B

Vaught, A. (2013). G95 compiler. In *web page*. https://g95.sourceforge.net