PGAS Languages -- An Easy-Entry Paradigm for Peta/ExaScale Computing

By Dinshaw S. Balsara (dbalsara@nd.edu) Physics and Applied Math Departments, University of Notre Dame, U.S.A.

Recent Advances: The last few years have seen the introduction of novel paradigms in parallel computing. The MPI-3 standard (Gropp *et al*. 2014) has made one-sided communication a practical reality. Language-based approaches to parallelism have been incorporated into the Fortran 2009 standard. These Fortran extensions go under the name of Coarray Fortran (CAF) and full-featured compilers that support CAF have become available from Cray and Intel; the GNU implementation is expected in 2015. There may also be movement towards crystallizing a UPC standard. The latter PGAS languages also excel at supporting one-sided communication, albeit in a simpler and more expressive format. Two questions remain: 1) ***Can they support a range of PetaScale applications?*** 2) ***Can they compete with the MPI-3 standard?*** In a recent paper (Garain, Balsara & Reid 2015) we have answered these questions in the affirmative and the present position paper provides a synopsis as well as identifies a way forward.

CAF combines elegance of expression with simplicity of implementation to yield an efficient parallel programming language. A video introduction to CAF is available from http://www.nd.edu/dbalsara/Numerical_PDE_Course. Elegance of expression results in very compact parallel code. It make CAF much easier to teach to students, thus enabling easy entry into high performance computing (HPC). Simplicity and ease of entry can also widen the base of entrants into Peta/ExaScale computing, thus hastening its widespread acceptance. The existence of a standard helps with portability and maintainability. CAF was designed to excel at one-sided communication and similar functions that support one-sided communication are also available in the recent MPI-3 standard. One-sided communication is expected to be very valuable for structured mesh applications involving partial differential equations, amongst other possible applications. This paper focuses on a comparison of CAF and MPI-3 for a few very useful applications areas that are routinely used for solving partial differential equations on structured meshes. The three specific areas are Fast Fourier Techniques, Computational Fluid Dynamics, and Multigrid Methods. We stress that we show results from production code that is used for science; these are not demonstrator applications.

For each of those applications areas, we have developed optimized CAF code and optimized MPI code that is based on the one-sided messaging capabilities of MPI-3. Weak scalability studies that compare CAF and MPI-3 are presented on up to 65,536 processors. Both paradigms scale well, showing that they are well-suited for Petascale-class applications. Some of the applications shown (like Fast Fourier Techniques and Computational Fluid Dynamics) require large, coarse-grained messaging. Such applications emphasize high bandwidth. Our other application (Multigrid Methods) uses pointwise smoothers which require a large amount of fine-grained messaging. In such applications, a premium is placed on low latency. Our studies show that both CAF and MPI-3 offer the twin advantages of high bandwidth and low latency for messages of all sizes. Even for large numbers of processors, CAF either draws level with MPI-3 or shows a slight advantage over MPI-3. Both CAF and MPI-3 have also been shown to provide substantial advantages over MPI-2. Fig. 1 shows the result of a weak scaling study using multigrid methods on NCSA's Blue Waters system. We show relative speedup of MPI-3 (red bars) and CAF (blue bars) as a function of increasing processor number. Fig. 2 shows a similar weak scaling study for a Computational Fluid Dynamics application.

Fig. 1 Relative scalability for 3-level Multigrid

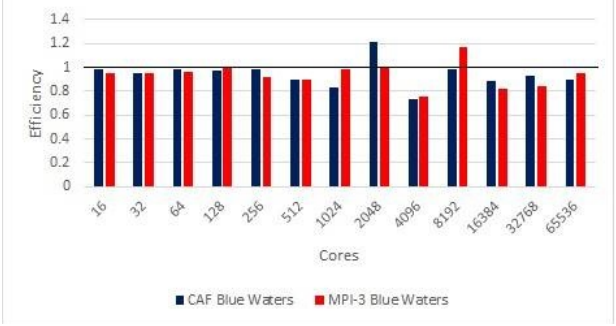
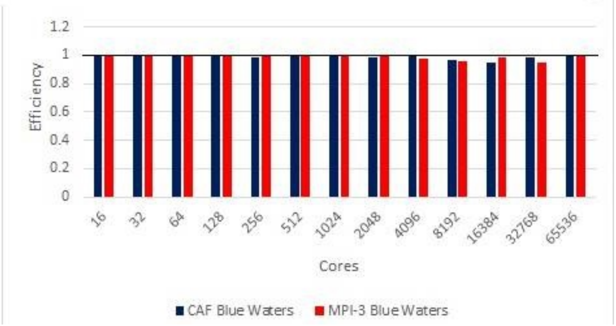


Fig. 2 Relative scalability for CFD Application



The Way Forward: Cray has developed network-in-a-card (NIC) technologies and NICs are expected to be available in high-end Xeon products from Intel. This technology has great potential for low-latency, energy-efficient messaging in future supercomputers. MPI-3 also has a prominent role to play because of two reasons: 1) The same one-sided programming style benefits MPI-3 and CAF. In fact, but for the messaging, our CAF and MPI-3 codes are identical. 2) On low-end HPC platforms, which lack NIC technologies, CAF compilers are built on top of MPI-3. In fact, this is the path taken by present generation GNU and Intel-based CAF compilers.

In addition to the weak scalability studies, Garain, Balsara & Reid (2015) have also catalogued some of the best-usage strategies that we have found for our successful implementations of one-sided messaging in CAF and MPI-3. We show that CAF code is of course much easier to write and maintain, and the simpler syntax makes the parallelism easier to understand, thereby making it easier for new entrants into Peta/ExaScale computing. Compilers that integrate CAF and OpenMP or CAF and OpenACC exist, further facilitating GPU usage. CAF operates on teams and/or sub-teams of processors and that feature can also be exploited (with modifications) to produce applications that are resilient to the failure of a few processors.

The author acknowledges support from NSF-ACI-1307369 and NSF-DMS-1361197.

References

S. Garain, D.S. Balsara & J.Reid, *Comparing Coarray Fortran (CAF) with MPI for Several Structured Mesh PDE Applications* ,submitted, Journal of Computational Physics, (2015)

W. Gropp, E. Lusk & A. Skjellum, *Using MPI - 3nd Edition: Portable Parallel Programming with the Message Passing Interface*, MIT Press (2014)