Validation and Verification Suite for OpenSHMEM

Swaroop Pophale¹, Tony Curtis¹, Barbara Chapman¹ and Stephen Poole²

¹ University of Houston Houston, Texas, USA
spophale,tonyc,chapman@cs.uh.edu
² Oak Ridge National Lab Oak Ridge, Tennessee, USA
spoole@ornl.gov

Abstract

With the evolution of many core and multicore systems different programming models have evolved to meet the needs of the current applications and their communication and computation needs. One such programming model is the Partitioned Global Address Space model which works to combine the advantages of both shared and distributed memory programming by providing a finer control over the data placement and access. OpenSHMEM is a PGAS library that allows for programmers to develop parallel SPMD applications with the help of its point-to-point (putget) communication operations, remote atomic memory operations, collective operations (like broadcast and reduce) and a simple set of ordering, locking, and synchronization primitives. The OpenSHMEM Specification provides general guidelines and expectation of the library’s behavior but without a validation suite the different vendor implementations could differ in their interpretation of the specification. In this paper we present the first Validation and Verification suite which not only validates an OpenSHMEM library’s functions but also provides micro-benchmarks that can be used to study and compare the performance of each of the individual library APIs for further analyses across different OpenSHMEM library implementations or different hardware configurations.

1 Introduction

In parallel computing over many and multi-core systems the Partitioned Global Address Space programming model has been largely successful in delivering optimum performance for distributed systems running SPMD applications. Popular language extensions and libraries under the PGAS programming model include Coarray Fortran (CAF), Unified Parallel C (UPC), OpenSHMEM etc. While CAF and UPC are language extensions, OpenSHMEM [2] is a library which provides a library API for one-sided point-to-point communication operations, remote atomic memory operations, collective operations (like broadcast and reduce) and a simple set of ordering, locking, and synchronization primitives. Till the OpenSHMEM specification V 1.0 [1] was finalized in 2011, the libraries were historically called SHMEM and had many variant flavors depending on the vendor and the targeted hardware system. These SHMEM library implementations vary in optimizations and hardware support they provide, but they all uphold the key concepts and principles of the original SHMEM library developed in 1993 by Cray Research Inc. Since the announcement of the specification, all SHMEM library implementations are striving to come together to support a unified API that will promote portability of applications and reproducibility of results across different (now) OpenSHMEM libraries that may run on different hardware platforms. To aid this effort University of Houston, along with Oak Ridge National Lab have developed the Validation and Verification (V&V) suite that checks for conformance of a given OpenSHMEM library to the current OpenSHMEM specification. In this poster we give an introduction to the OpenSHMEM specification and the key concepts of
OpenSHMEM which are the foundation of the Validation and Verification suite. We discuss the different OpenSHMEM library calls and the expected behavior of an implementation. We also deliberate over the different ways in which we verify that an OpenSHMEM library adheres to the specification and the challenges that we faced therein.

References

[1] Openshmem specification 1.0 (openshmem.org).