Validation and Verification of OpenMP Offloading & OpenACC Compilers on UD DARWIN System

UDEL Team: Aaron Jarmusch, Nolan Baker, Nikhil Rao, Christian Munley, Vaidhyanathan Ravichandran, Michael Carr, Jaydon Reap, Kristina Holsapple, Aaron Liu, Khai McCaskill, Daniel Horta, Felipe Cabarcas, Sunita Chandrasekaran
Collaborators: OpenACC, NVIDIA, Oak Ridge National Laboratory, DOE Exascale Computing Project

ABSTRACT

The **OpenMP** and **OpenACC** language specifications continue to evolve, and with each new release, new features are introduced. The need to validate and verify these new features is crucial. The SOLLVE OpenMP Validation and Verification team as well as the OpenACC Validation and Verification team focus on:

- Evaluating conformity & implementation of OpenMP on compilers, including:
  - GNU's GCC, Clacc/LLVM, Intel's ICC, AMD's ROCm, and NVIDIA's HPC SDK
- Simularity of OpenACC on compilers, including:
  - GNU's GCC, Clacc/LLVM, and NVIDIA's HPC SDK
- Validating implementation of compiler builds across HPC systems.

This validation and verification process is being run on the **DARWIN system**.

BACKGROUND

OpenMP and OpenACC are parallel-based programming models which allow for performance optimization in **C, C++ & Fortran**, with its features called **"directives"** listed in their respective specifications (spec). The purpose of the Validation and Verification (V&V) testsuites are to:

- Evaluate compiler's **compliance** with the specification
- **Identify** ambiguities in the specification
- Illustrate a system's ability to run directives & utilize offloading parallel directives on GPUs
- **Demonstrate** the use & purpose of new directives to application developers

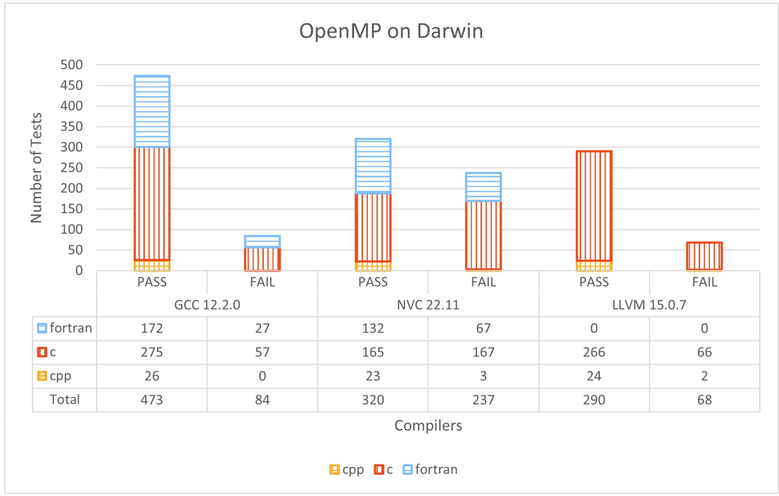
OpenMP and OpenACC are useful for many application developers working on HPC systems to ensure their code is running at maximum efficiency. These testsuites ensure that the respective specifications, compiler vendors & system operators are implementing OpenMP and OpenACC effectively.

```
  for (int dev = 0; dev < num_dev; ++dev) {
#pragma omp target data map(tofrom: array[0:N]) device(dev)
  {
  #pragma omp target map(alloc: array[0:N]) device(dev)
  {
    for (int i = 0; i < N; ++i) {
      array[i] += dev + 1;
    }
  } // end of omp target
  } // end of omp target data
  for (int i = 0; i < N; ++i) {
  OMPVV_TEST_AND_SET(errors, array[i] != dev); }
```

Fig. 1: Code segment showing test for target device directive

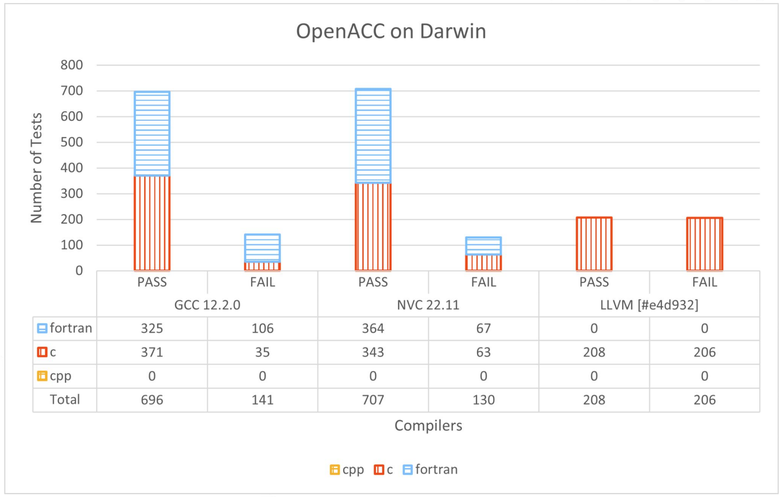
Figure 1 shows a test segment from the SOLLVE testsuite, where the array values are set to the device number. If the dev variable in the target region is not equal to dev variable set outside of the target region, then there will be errors. Otherwise, the test will pass.

OpenMP & OpenACC RESULTS



	GCC 12.2.0		NVC 22.11		LLVM 15.0.7	
	PASS	FAIL	PASS	FAIL	PASS	FAIL
fortran	172	27	132	67	0	0
c	275	57	165	167	266	66
cpp	26	0	23	3	24	2
Total	473	84	320	237	290	68

Fig. 2: Graph showing OpenMP V&V results on UD's DARWIN system.



	GCC 12.2.0		NVC 22.11		LLVM [#e4d932]	
	PASS	FAIL	PASS	FAIL	PASS	FAIL
fortran	325	106	364	67	0	0
c	371	35	343	63	208	206
cpp	0	0	0	0	0	0
Total	696	141	707	130	208	206

Fig. 3: Graph showing OpenACC V&V results on UD's DARWIN system.

OPEN SOURCE REPOSITORIES

Our suites are available on GitHub for download, logging issues, suggesting tests, etc.

OpenMP SOLLVE

OpenACC V&V

DISCUSSION

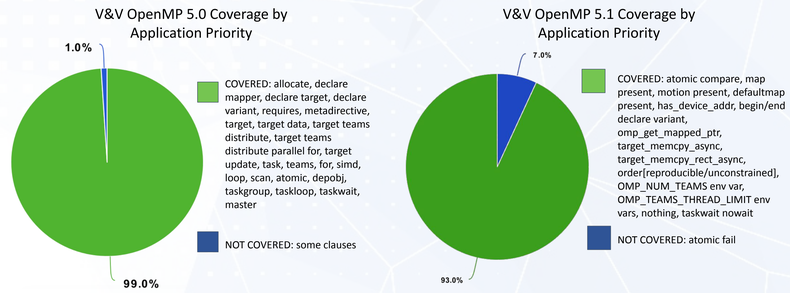


Fig. 4: Pie charts showing OpenMP V&V coverage of OpenMP 5.0 and 5.1 specifications.

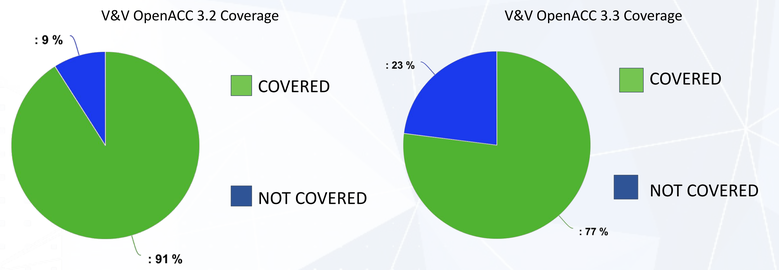


Fig. 5: Pie charts showing OpenACC V&V coverage of OpenACC 3.2 and 3.3 specifications.

FUTURE

- Working on new tests for the latest OpenMP specification (5.2) and OpenACC specification (3.3)
- Expand our Fortran tests so that more C/C++ tests have Fortran counterparts
- OpenACC Example Guide and Practice–codes to help users understand the specification
- Running/Testing the suite on more machines to further test target devices
  - Crusher, Perlmutter, Summit, Sunspot

ACKNOWLEDGEMENTS

This research was supported in part through the use of DARWIN computing system: DARWIN – A Resource for Computational and Data-intensive Research at the University of Delaware and in the Delaware Region, which is supported by NSF under Grant Number: 1919839.
This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.
This research was supported by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration.

Contact US: <jarmusch, nolanb> @udel.edu crpl.cis.udel.edu