Using Jetstream for OpenMP Offloading and OpenACC Testsuites

Aaron Jarmusch, Nolan Baker
Sunita Chandrasekaran
{jarmusch, nolanb, schandra, University of Delaware}

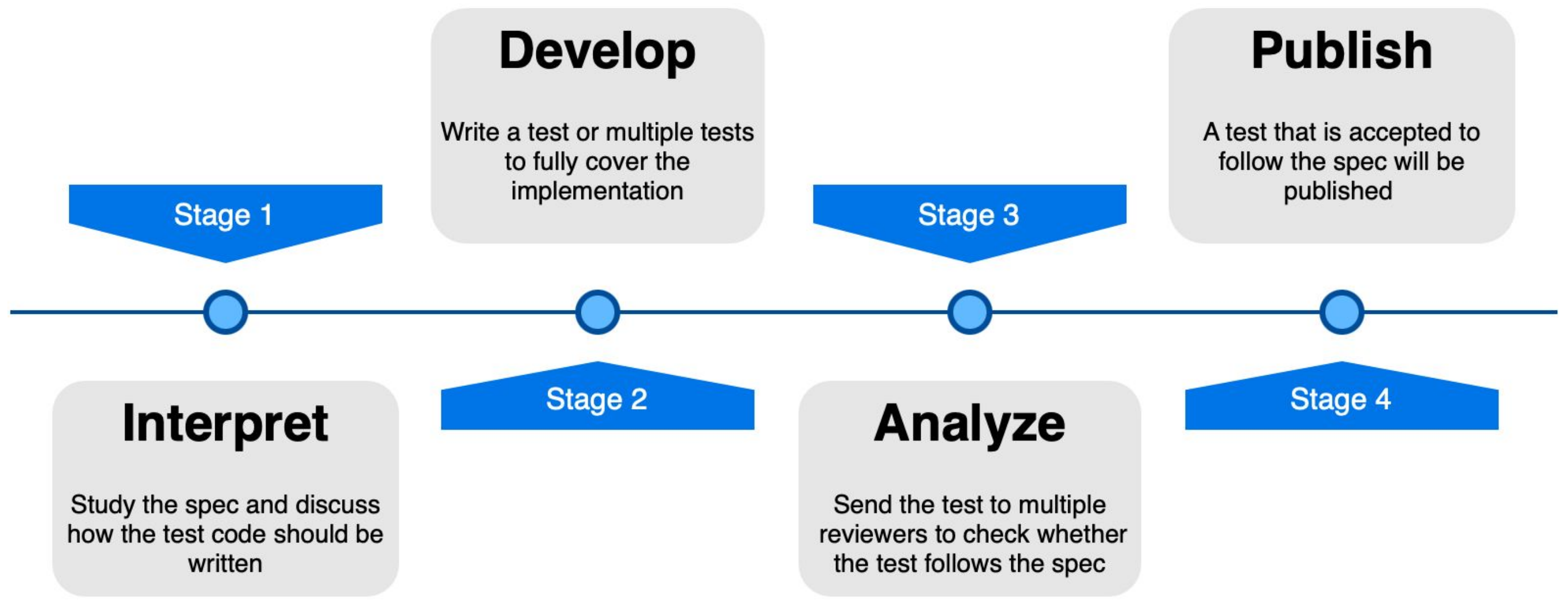
Introduction

- **OpenMP** and **OpenACC**
  - Directive-based programming models that target heterogeneous systems.
  - Specifications are evolving as we speak to accommodate the needs of application developers.
  - This project focuses on the **OpenMP SOLLVE** & **OpenACC Validation and Verification testsuites** that evaluate compilers' compliance with the specification and identifies ambiguities in the specification.

Project Goals

1. Create **unit tests** based on the specification of the directives.
2. Discuss whether bugs are caused by **Specification issues** or an **Implementation bug**.
3. Report bugs to OpenMP/OpenACC or vendors respectively.
4. Organize & display tests' compiler & runtime **outcome**, on systems where the tests ran.

Approach


Develop
Write a test or multiple tests to fully cover the implementation
Publish
A test that is accepted to follow the spec will be published
Stage 1
Stage 3
Stage 2
Stage 4
Interpret
Study the spec and discuss how the test code should be written
Analyze
Send the test to multiple reviewers to check whether the test follows the spec

Flowchart displaying process of writing OpenMP/ACC tests.

Writing Tests:

- Specifications, which narrates definitions of features and their clauses that are to be analyzed.
- Tests are written which would not pass if the directive does not function correctly.

Test Analysis:

- Has the directive in the test adhered to the definition in the specification?
- If the test passes, a pull request (PR) to the testsuite is made approved by another team member.
- If it fails, bug reports are sent to vendor.
- Rare case of unclear or misleading directive definitions are discussed with the organizations.

Results

Jetstream Setup:

- Processor: 1-8 Intel(R) Xeon(R) Gold 6130 CPU @ 2.10Ghz, 47GB RAM
- Accelerator: GRID V100X-16Q
- Operating System: Ubuntu 18

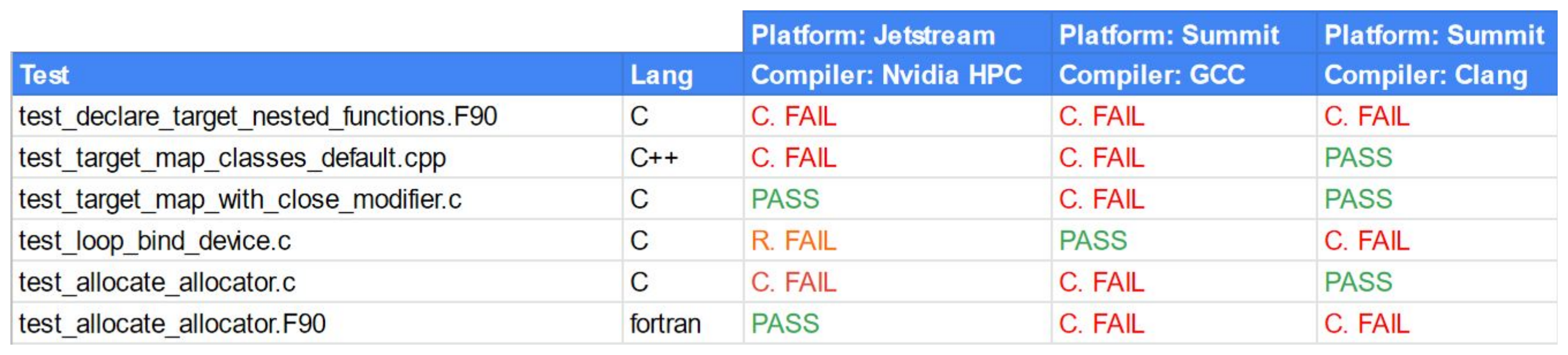
Results on Jetstream:

Compiler: Nvidia NVHPC SDK 21.5	Jetstream
Testsuite	Pass/Fail - Total
OpenMP Version 4.5	162/49 - 211
OpenMP Version 5.0	72/115 - 187
OpenACC	683/134 - 817

This table displays results from both OpenMP & OpenACC on the **XSEDE Jetstream** HPC system.

- These tests use the **Nvidia NVHPC SDK** suite with the OpenACC and OpenMP compilers.
- As shown in the table the OpenMP SOLLVE V&V is comprised of two versions while the OpenACC V&V is an all inclusive testsuite of each OpenACC version.

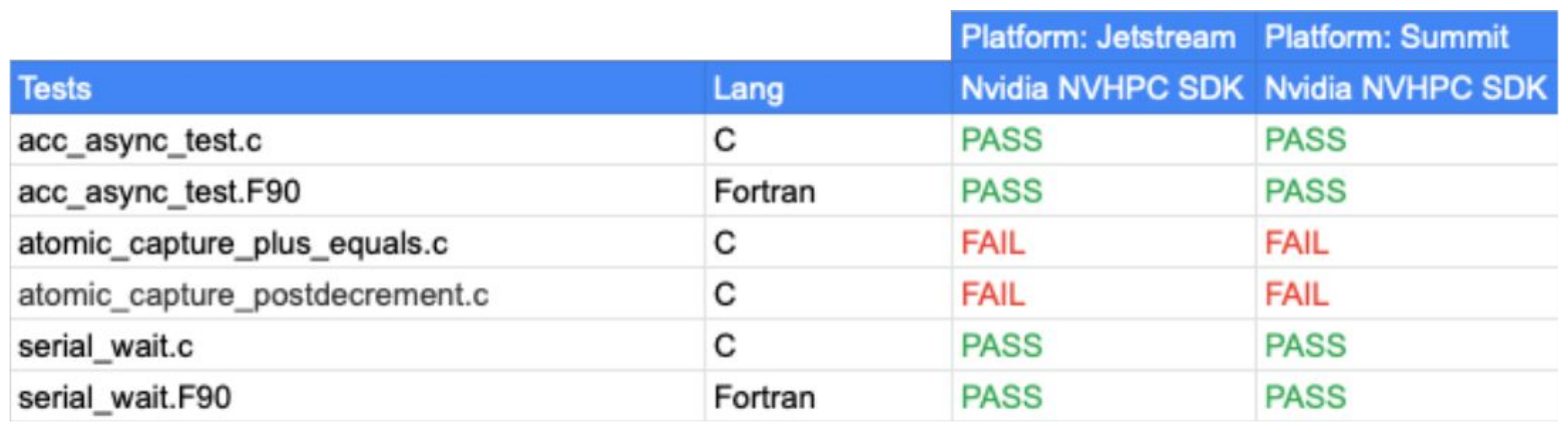
OpenMP



		Platform: Jetstream	Platform: Summit	Platform: Summit
Test	Lang	Compiler: Nvidia HPC	Compiler: GCC	Compiler: Clang
test_declare_target_nested_functions.F90	C	C. FAIL	C. FAIL	C. FAIL
test_target_map_classes_default.cpp	C++	C. FAIL	C. FAIL	PASS
test_target_map_with_close_modifier.c	C	PASS	C. FAIL	PASS
test_loop_bind_device.c	C	R. FAIL	PASS	C. FAIL
test_allocate_allocator.c	C	C. FAIL	C. FAIL	PASS
test_allocate_allocator.F90	fortran	PASS	C. FAIL	C. FAIL

Examples of OpenMP and OpenACC test pass/fail table. Tests can either pass (PASS), compilation fail (C.FAIL) or runtime fail (R.FAIL).

OpenACC



		Platform: Jetstream	Platform: Summit
Tests	Lang	Nvidia NVHPC SDK	Nvidia NVHPC SDK
acc_async_test.c	C	PASS	PASS
acc_async_test.F90	Fortran	PASS	PASS
atomic_capture_plus_equals.c	C	FAIL	FAIL
atomic_capture_postdecrement.c	C	FAIL	FAIL
serial_wait.c	C	PASS	PASS
serial_wait.F90	Fortran	PASS	PASS

Tables show importance of not only the compiler, but the system architecture it is running on.

More Results found here

Test Format

OpenACC Code Example:

```
#pragma acc data copyin(a[0:n])
      #pragma acc parallel copyout(b[0:n])
          #pragma acc loop
          for (int x = 0; x < n; ++x)
              b[x] = a[x];
```

OpenMP Code Example:

```
#pragma omp parallel for
  for(int i=1; i<N; i++){
    #pragma omp task depend(inout:x) shared(x)
    x=i;
    #pragma omp task depend(inout: y) shared(y)
    y=i;
    #pragma omp taskwait depend(in:x)
    OMPVV_TEST_AND_SET(errors, x != i);
    #pragma omp taskwait depend(in: x,y)
    OMPVV_TEST_AND_SET(errors, y!= i && x!= i);
  }
```

Discussions

- OpenMP on Jetstream was tested only on the **host** device, while OpenACC tested with **GPU offloading**.
- Nvidia's NVHPC SDK suite of compilers are more compatible with the OpenACC spec compared to the OpenMP spec.
- Based on the results & installation process, OpenACC is easier to use on the Jetstream system.

Future Work

- While the results in this poster are focusing on IU's Jetstream, the OpenMP SOLLVE V&V Testsuite is also running on **OakRidge National Laboratorys** Summit, Spock, Tulip, and Cori HPC systems.
- Work in progress entails writing test cases for OpenMP 5.1 and OpenACC 3.1

References

This material is based upon work supported by the National Science Foundation under Grant No. 1445604 and 1548562. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

1. Jose Monsalve Diaz, Kyle Friedline, Swaroop Pophale, Oscar Hernandez, David E. Bernholdt, Sunita Chandrasekaran, Analysis of OpenMP 4.5 Offloading in Implementations: Correctness and Overhead, Parallel Computing, Volume 89, 2019, 102546, ISSN 0167-8191, https://doi.org/10.1016/j.parco.2019.102546.
2. Diaz J.M., Pophale S., Hernandez O., Bernholdt D.E., Chandrasekaran S. (2018) OpenMP 4.5 Validation and Verification Suite for Device Offload. In: de Supinski B., Valero-Lara P., Martorell X., Mateo Bellido S., Labarta J. (eds) Evolving OpenMP for Evolving Architectures. IWOMP 2018. Lecture Notes in Computer Science, vol 11128. Springer, Cham. https://doi.org/10.1007/978-3-319-98521-3_6
3. Jose Monsalve Diaz, Swaroop Pophale, Kyle Friedline, Oscar Hernandez, David E. Bernholdt, and Sunita Chandrasekaran. 2018. Evaluating Support for OpenMP Offload Features. In Proceedings of the 47th International Conference on Parallel Processing Companion (ICPP '18). Association for Computing Machinery, New York, NY, USA, Article 31, 1–10. DOI:https://doi.org/10.1145/3229710.3229717
4. C. Wang, R. Xu, S. Chandrasekaran, B. Chapman and O. Hernandez, "A Validation Testsuite for OpenACC 1.0," *2014 IEEE International Parallel & Distributed Processing Symposium Workshops*, 2014, pp. 1407-1416, doi: 10.1109/IPDPSW.2014.158.
5. Friedline K., Chandrasekaran S., Lopez M.G., Hernandez O. (2017) OpenACC 2.5 Validation Testsuite Targeting Multiple Architectures. In: Kunkel J., Yokota R., Taufer M., Shalf J. (eds) High Performance Computing. ISC High Performance 2017. Lecture Notes in Computer Science, vol 10524. Springer, Cham. https://doi.org/10.1007/978-3-319-67630-2_3