OpenMP Validation and Verification (V&V) Testsuite

Sunita Chandrasekaran
Associate Professor, University of Delaware, USA

Computational Scientist, Brookhaven National Lab, USA
Contact: schandra@udel.edu

SIAM PP MS64 - Testing and Verification for Performance Portable Programming Systems

V&V Current Team

CRPL

OAK RIDGE National Laboratory

- Andrew Kallai
- Felipe Cabarcas[1]
  - (massive shoutout for all the plots in this talk)
- Sunita Chandrasekaran
- Swaroop Pophale
- Seyong Lee
- David E. Bernholdt

- Outside the project contributors
  - AMD
  - Tobias Schuele (Siemens) for his feedback and engagement with the tests
- Former project members
  - Nolan Baker, Michael Carr, Nikhil Rao, Jaydon Reap, Kristina Holsapple, Joshua Hoke Davis, Thomas Huber, Jose M. Monsalve

Huber, T., Pophale, S., Baker, N., Carr, M., Rao, N., Reap, J., ... & Chandrasekaran, S. ECP SOLLVE: validation and verification testsuite status update and compiler insight for openMP. In 2022 IEEE/ACM International Workshop on Performance, Portability and Productivity in HPC (P3HPC) (pp. 123-135), (2022, November). IEEE.

[1] UNIVERSIDAD DE ANTIOQUIA

Goal

Create unit tests to validate & verify compilers' implementation of the specification

Revealing ambiguities in the specification

Determining missing implementation of a feature

Highlighting unmentioned restriction of a feature

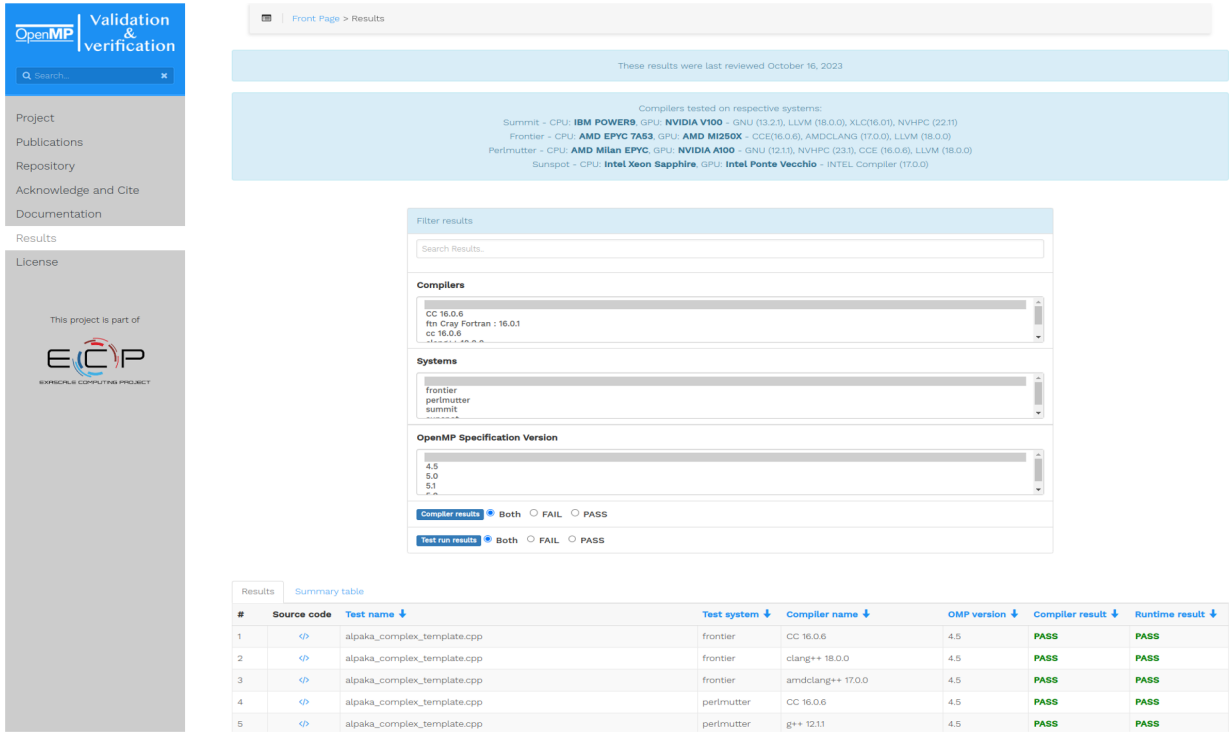
Evaluating implementations for multiple target platforms

Identifying and reporting compiler bugs

V&V Objectives

- Tests implementations of new features introduced in OpenMP 4.5+
- Checks conformance to the specification
  - Peer reviewed, publicly available
- Highlights ambiguities in the OpenMP specification
- Reports status of implementations on primary ECP target platforms
  - Help application and compiler developers
- Exemplifies the use of the new features in OpenMP
- Abstracts application kernels as tests

Our results website



https://crpl.cis.udel.edu/ompvvsollve/results/

Latest OpenMP Specification: 5.2

- Released November 2021
  - Around 27 modifications (new features, deprecated features, behavior changes)
  - Some of these modifications to the specification are implemented in open source compilers
    - LLVM status https://clang.llvm.org/docs/OpenMPSupport.html
    - GCC status https://gcc.gnu.org/wiki/openmp
  - Newer tests to be written
  - But are there implementations to test with? Who are the users of these features? Is there a priority list of which feature should be implemented?
- 6.0 specification is scheduled to be released on November 2024

OpenMP V&V Testsuite repository

- Clone the repo https://github.com/SOLLVE/sollve_vv
- Setup your environment (install or 'module load' your compilers)
- Use our make commands to compile and run a single test or the whole suite
  - Running a single test:

```
make CC=clang CXX=clang++ FC=flang-new OMP_VERSION=5.2 SOURCES=test_name all
```

  - Running the entire suite

```
make CC=clang CXX=clang++ FC=flang-new OMP_VERSION=5.2 all
```

Tests and Coverage

Number of Tests	C	C++	Fortran	Total
4.5	121	14	104	**239**
5.0	190	13	128	**331**
5.1	96	1	26	**123**
5.2	15	9	4	**28**
Total	**422**	**37**	**262**	**721**

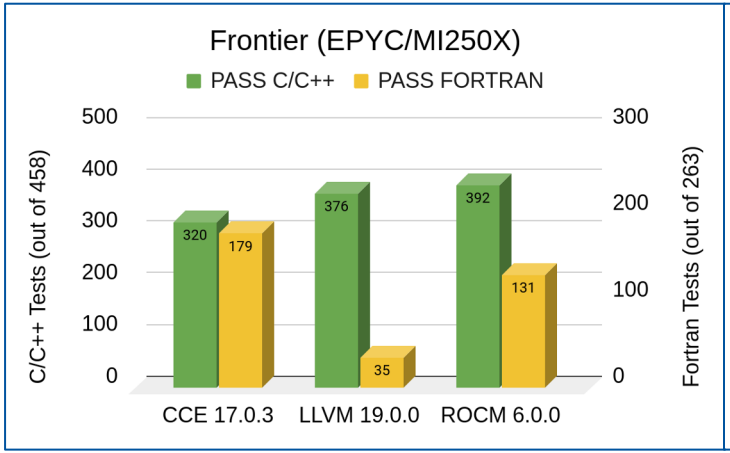
Approximated coverage of requested features	C	C++	Fortran
4.5	100%	100%	100%
5.0	88%	100%	88%
5.1	87%	100%	40%
5.2	84%	100%	19%

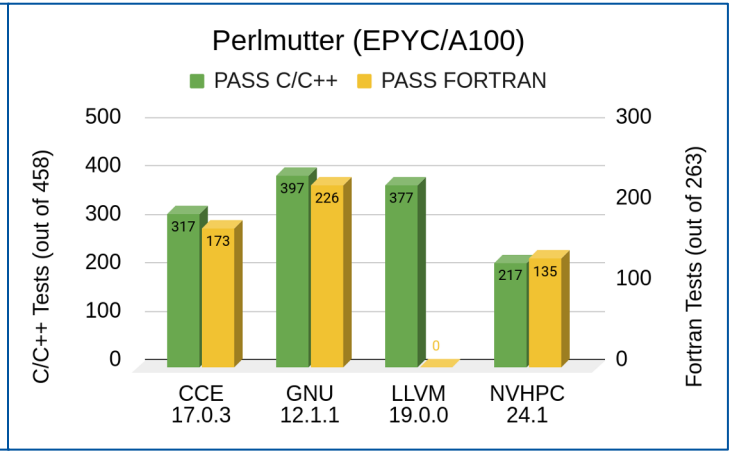
- The testsuite not only contains tests for the new features of each specification, but also commonly used features combinations by application developers
- We have given priority to requested features over complete coverage!!!!

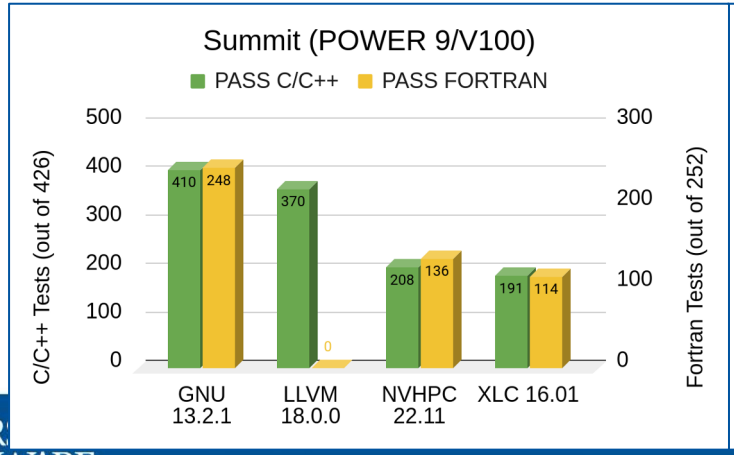
Systems and compilers regularly tested

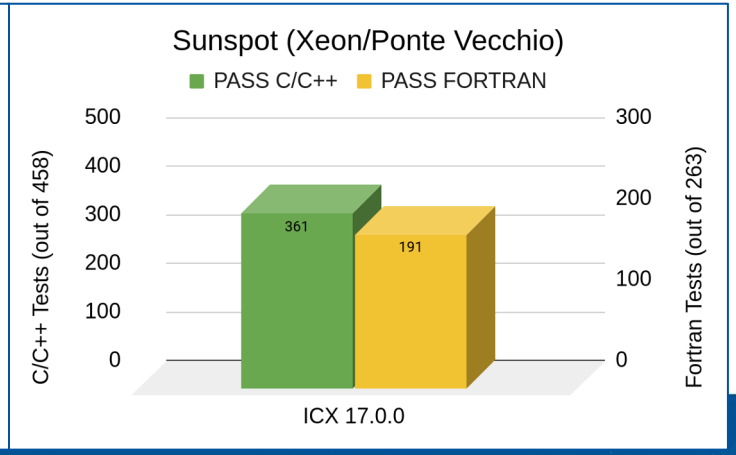
System	*Perlmutter*	*Summit*	*Frontier*	*Sunspot*
Location	*NERSC*	*ORNL*	*ORNL*	*ANL*
CPU / GPU	*AMD / NVIDIA*	*IBM / NVIDIA*	*AMD / AMD*	*Intel / Intel*
GNU (gcc/gfortan)	YES	YES	NO	NO
LLVM (clang/flang-new)	YES	YES	YES	NO
AMD ROCM (amdclang/amdflang)	NO	NO	YES	NO
HPE/Cray (cce)	YES	NO	YES	NO
IBM (xlc, xlf)	NO	YES	NO	NO
NVIDIA (nvc, nvfortran)	NO	YES	YES	NO
INTEL (icpx, ifx)	NO	NO	NO	YES

Overall Results

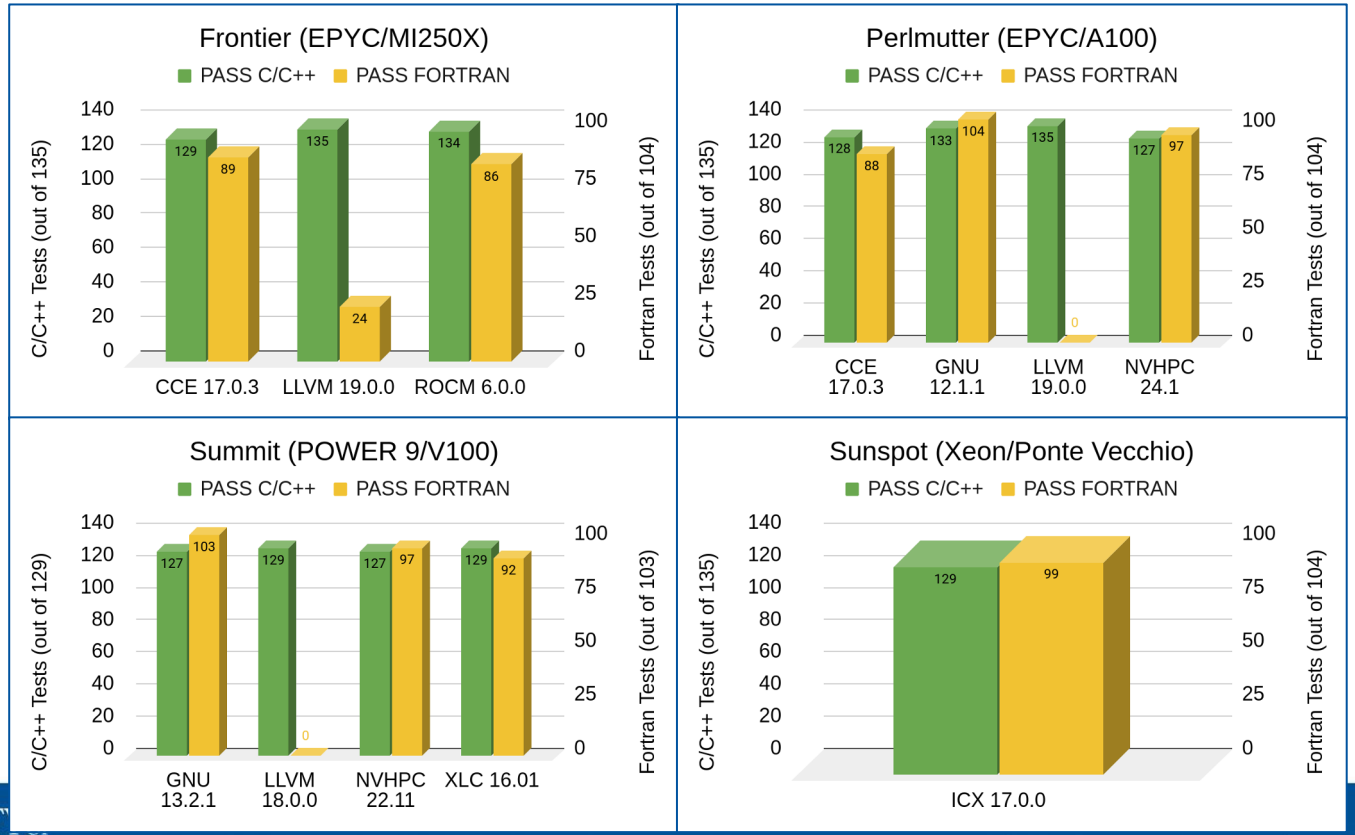




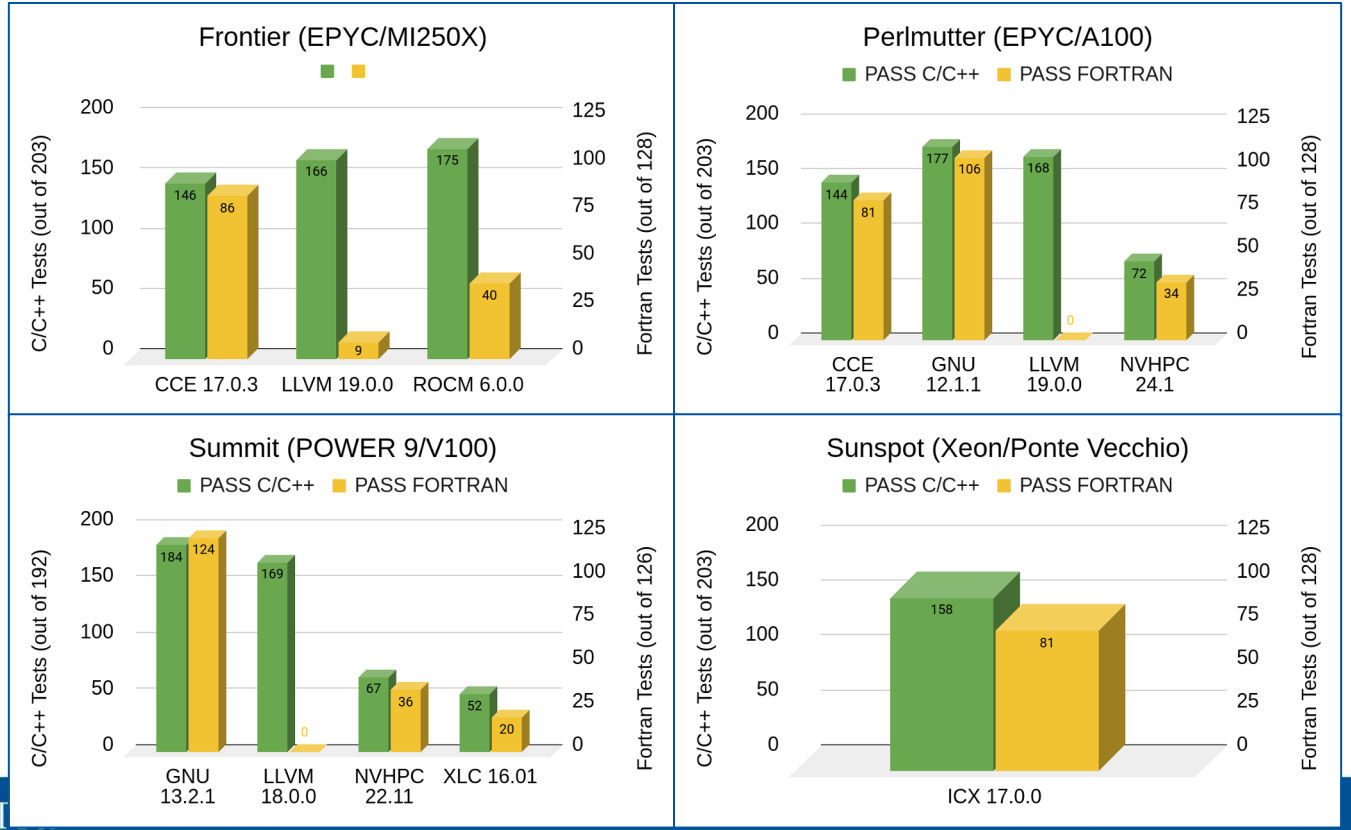




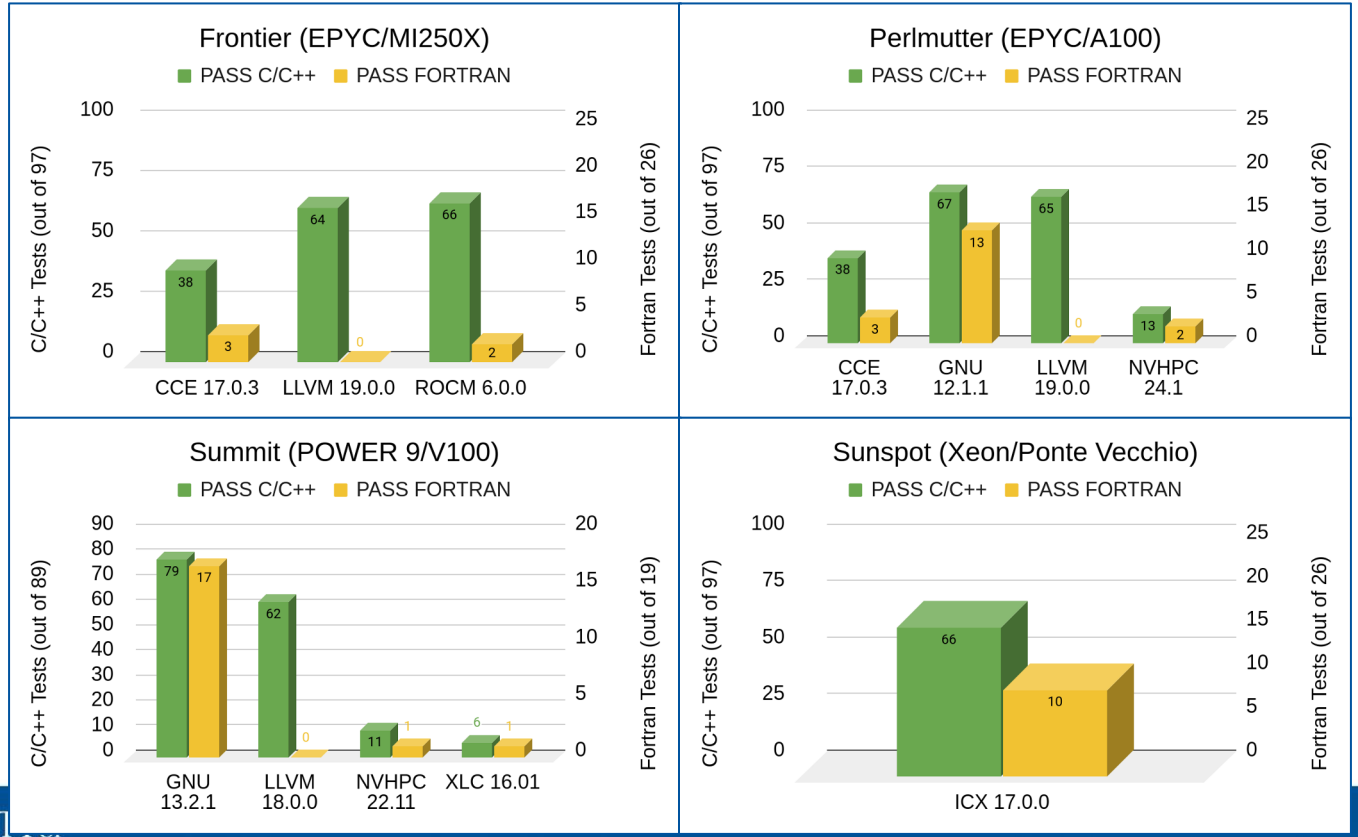
OpenMP 4.5 results



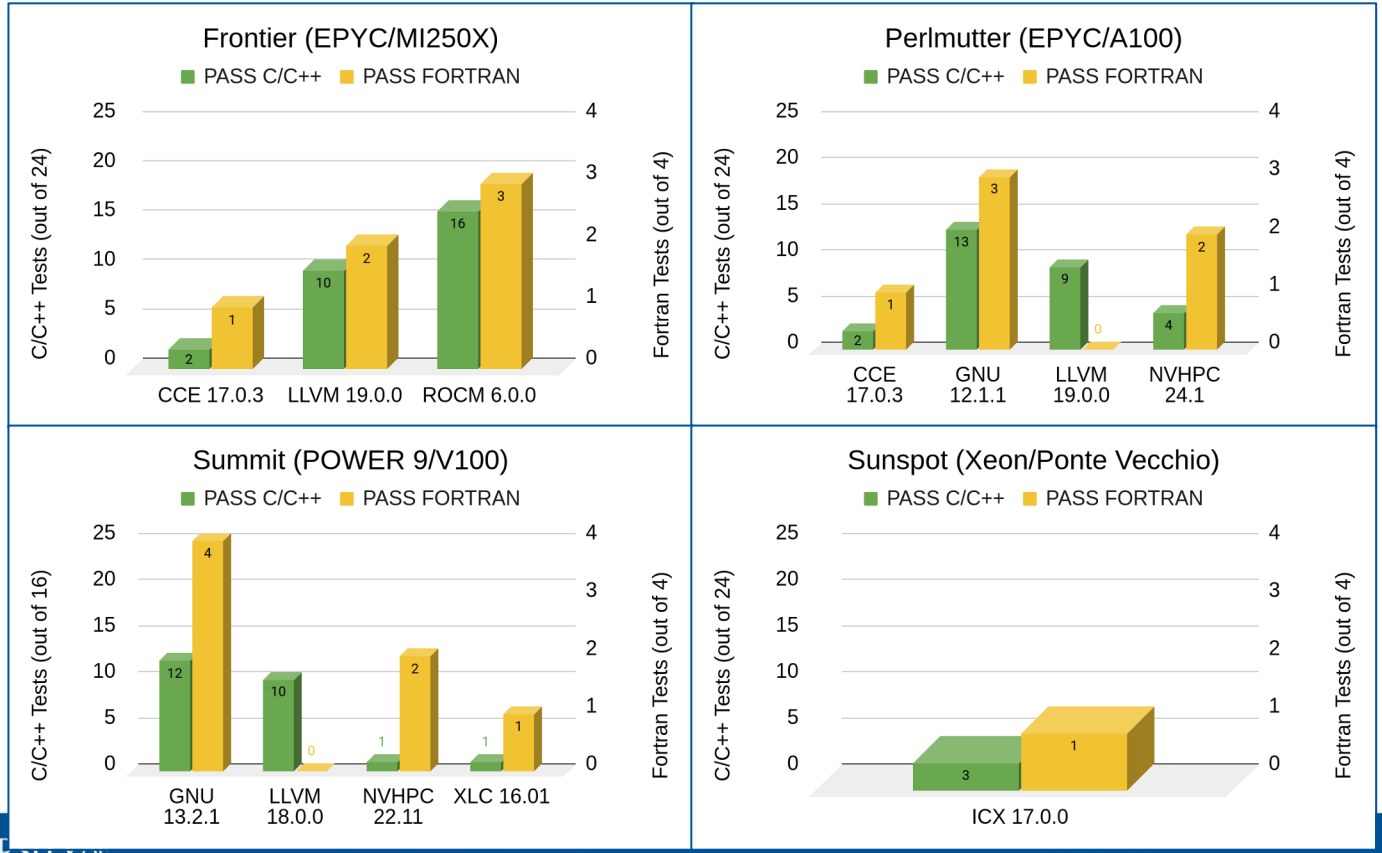
OpenMP 5.0 results



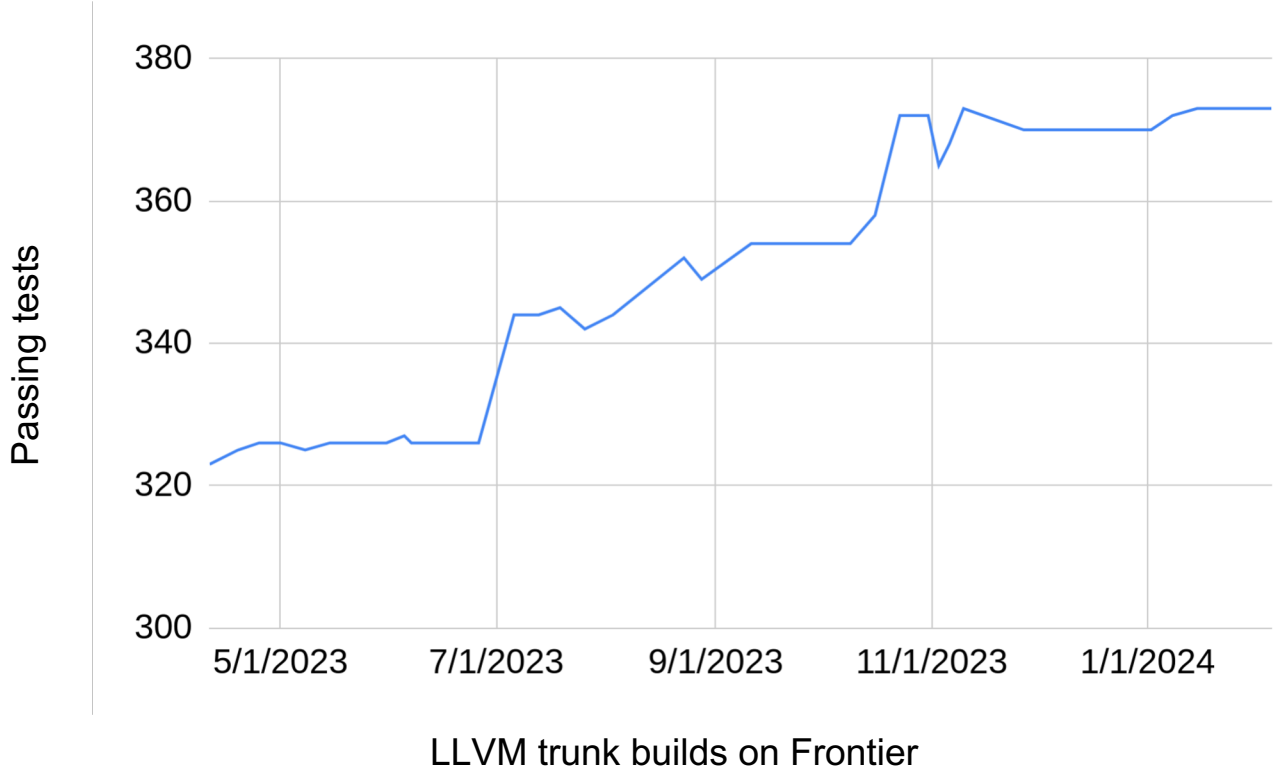
OpenMP 5.1 results



OpenMP 5.2 results



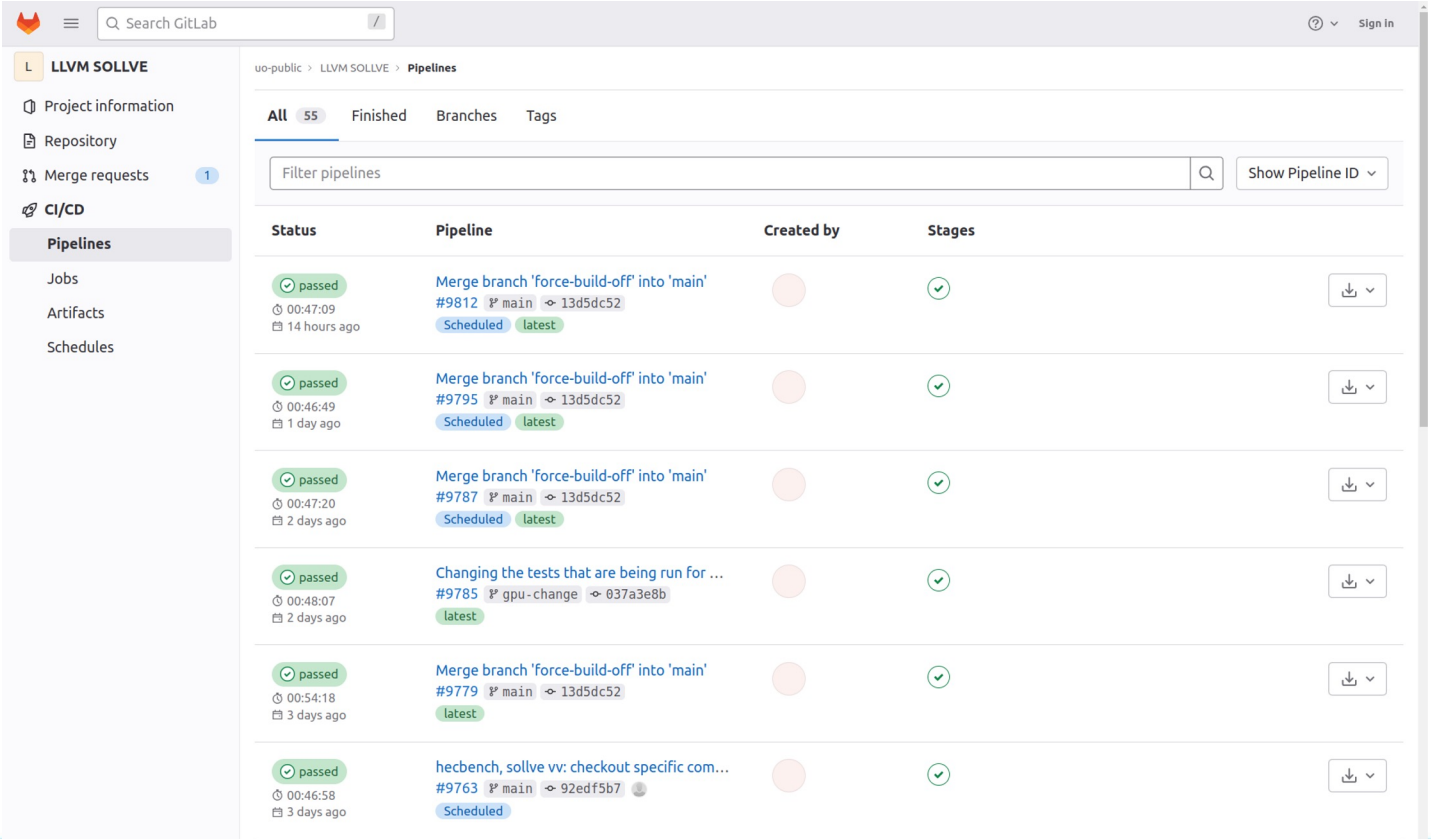
Evolution of LLVM OpenMP offloading (Frontier system)



CI/CD - V&V

Extreme scale scientific software stack (E4S)

- Integrating regression and stress testing into the V&V testsuite
- Creating a CI infrastructure to test LLVM OpenMP on GPU systems
  - Running V&V, Hecbench and AMD's Smoke
  - Focusing on offloading for AMD and NVIDIA GPUs
    - https://gitlab.e4s.io/uo-public/llvm-sollve/-/pipelines
    - Using University of Oregon cluster
- LLVM Nightly Testing: a set of client and server tools for monitoring the performance of software over its lifecycle
  - http://lnt.llvm.org/



https://gitlab.e4s.io/uo-public/llvm-sollve/-/pipelines

Links to resources

- GitHub for V&V tests
  - https://github.com/SOLLVE/sollve_vv
- Most recent paper published with P3HPC @ SC22
  - https://arxiv.org/abs/2208.13301
- Results page
  - https://crpl.cis.udel.edu/ompvvsollve/
- OpenMP example guides for each specification on the OpenMP site
  - https://www.openmp.org/specifications/

Generative AI for validation
LLM4VV work in progress

based on OpenACC V&V testsuite

- Standard specification evolves
- Programmers must learn and adapt: regular development
- Could we use LLMs to automate?
- Prompts? Fine-tune? Train new LLMs?
- How to determine the quality of the LLM-generated tests?
- How do we tackle “hallucinations”?
- Watch out for carbon footprint when training LLMs
  - GPT-4 is the equivalent of powering more than 1300 homes for one year

Massive shoutout:
Christian Munley (senior undergrad for LLM work) and Aaron Jarmusch – 1st year PhD student (for OpenACC V&V work)

Generative AI for validation

- Star
- Prog
- Cou
- Pro
- Hov
- Hov
- Wat
  - GPT-4 is the equivalent of powering more than 1300 homes for one year

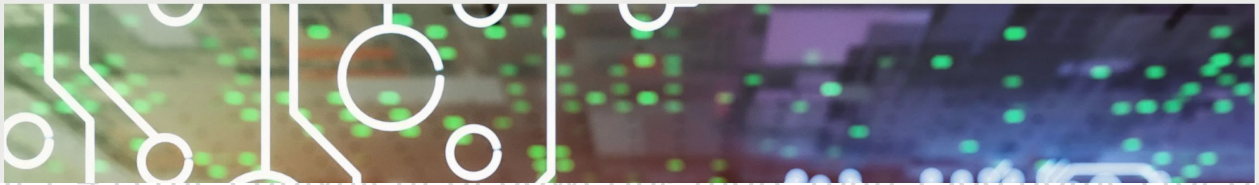
IEEE Spectrum / AI Prompt Engineering Is Dead

Type to search

NEWS | ARTIFICIAL INTELLIGENCE

AI Prompt Engineering Is Dead › Long live AI prompt engineering

BY DINA GENKINA | 06 MAR 2024 | 6 MIN READ |



LLM4VV: Observations

- Doesn’t get the correct feature
- Error with base language syntax/semantics
- Error with an implementation
- Incorrect testing logic
- Unimplemented features (compiler/runtime/non-zero error)
- Not testing the intended feature (no isolation)
- Hallucinations

State of the art - Evaluation of LLMs against relevant code-based benchmarks

- Choice of LLMs

LLMs / Benchmark	HumanEval+ pass@1	MBPP+ pass@1	GSM8K	TruthfulQA
Codelama-34B-Instruct	43.9 [36]*	52.9 [36]*	32.7 [37]*	47.4 [37]
Phind-Codellama-34b-v2	67.1 [36]	-	-	-
Deepseek-Coder-33b-Instruct	75.0 [36]	66.7 [36]	60.7 [38]*	-
GPT-3.5(-Turbo)	70.7 (turbo) [36]	69.7 (turbo) [36]	57.1 [1]	47.0 [1]
GPT-4(-Turbo)	**81.7** (turbo) [36]	**70.7** (turbo) [36]	**92.0** [1]	**60.0** [1]

Table 1: Performance of LLMs using relevant benchmarks in percentages. The metric is pass@k metric that indicates unit test pass rate when selecting respectively k samples from the candidate solutions [39]. The pass@k metric for GSM8k and TruthfulQA seemed unclear. We could not find the results for all LLMs and benchmarks. We have cited references for each value. Results for some of these LLMs are for the base LLM as indicated in superscript *.

LLM4VV: Evaluation of LLMs, their fine-tuned versions and prompt engineering methods

- Deepseek-Coder-33b-Instruct generated most passing tests and in some cases GPT-4 Turbo – work in progress

LLMs / Methods	Template	Template + RAG	Oneshot	Oneshot + RAG	Expressive + Template + RAG
Codellama-34b-Instruct	15	11	7	4	35
Phind-Codellama-34b-v2	48	43	52	22	43
Deepseek-Coder-33b-Instruct	**51**	**56**	51	**57**	47
GPT-3.5-Turbo	15	19	5	11	23
GPT-4-Turbo	48	45	**54**	49	**54**
Fine-tuned GPT-3.5-Turbo					27
Fine-tuned Phind					44
Fine-tuned Deepseek					47

Table 2: Stage 1 results displaying passing percentage for all selected LLMs and five methods. We see that Deepseek-Coder-33b-Instruct generally produced the most passing tests, though in some cases GPT-4-Turbo produced more - these are indicated **in bold**. We do not have results for the Fine-tuned GPT-3.5-Turbo, Phind and Deepseek LLMs because our focus is on exploring the Fine-tuning method of these LLMs against prompt-based methods rather than "with" the prompt-based methods. However we did run the three Fine-tuned LLMs with Expressive + Template + RAG prompts as we found "with" this prompt-based methods, the non-Fine-tuned LLMs achieved the best performance.

Experimental Setup

LLMs	Tests generated	GPU Time Taken	GPUs used	Model Parameter
Codellama-34B-Instruct	351	~3.6 hours	4 A100s	34B
Phind-Codellama-34B-v2	351	~3.6 hours	4 A100s	34B
Deepseek-Coder-33b-Instruct	351	~3.6 hours	4 A100s	33B
GPT-4-Turbo	351	~4 hours	Unknown	Unknown
Fine-tuned GPT-3.5-Turbo	351	~4 hours	Unknown	174B
Fine-tuned Phind-Codellama-34B-v2	351	~3.6 hours	4 A100s	34B
Fine-tuned Deepseek-Coder-33b-Instruct	351	~3.6 hours	4 A100s	34B

Table 3: Experimental setup for Stage 2 using the LLMs, including the number of tests generated, GPUs used and the time they took for inference

LLM4VV: To be Contd

- Manual analysis of automatically generated tests
- Gaps in LLM, programming model, and base languages
- With best LLMs, various prompts, context from specification, they fail often
- Developer speedups for some use cases. Always requires oversight
- Challenge - metric for accuracy of test beyond human analysis?
- Future works will include further training, iterative prompting, and human-in-the-loop
- Question: can we access the specification latex for this research?

Aaron Jarmusch, Christian Munley, Sunita Chandrasekaran, LLM4VV: Developing LLM-Driven Testsuite for Compiler Validation, Under review - https://arxiv.org/abs/2310.04963

Summary

- Could LLMs create a brand new test from a specification?
- LLMs are fun to explore but need to be cautious
- Contributions (new tests or PRs) and feedback to the V&V – most welcome!
- Teaching and training the next-generation workforce is only becoming increasing a challenge! (a pleasant one perhaps :))