Cray User Group Meeting 2024 (CUG 2024)
May 5-9th, 2024

Argonne NATIONAL LABORATORY

Early Application Experiences on Aurora at ALCF

Moving From Petascale to Exascale Systems

Colleen Bertoni*, JaeHyuk Kwack*, Thomas Applencourt, Abhishek Bagusetty, Yasaman Ghadar, Brian Homerding, Christopher Knight, Ye Luo, Mathialakan Thavappiragasam, John Tramm, Esteban Rangel, Umesh Unnikrishnan, Timothy J. Williams, Scott Parker

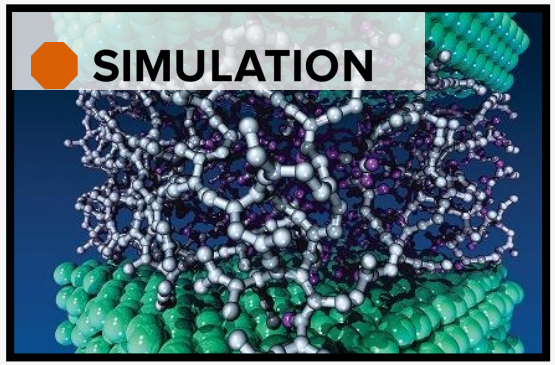
Argonne Leadership Computing Facility

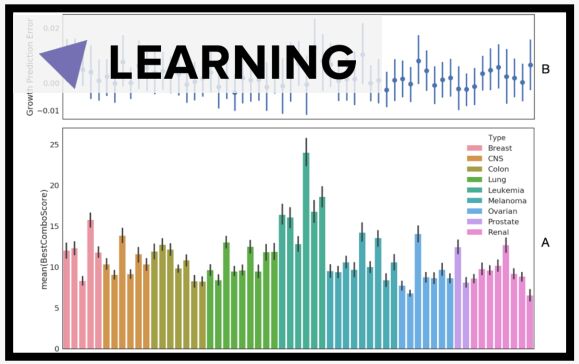
The Argonne Leadership Computing Facility

Argonne Leadership Computing Facility

The Argonne Leadership Computing Facility provides world-class computing resources to the scientific community.

- Users pursue scientific challenges
- In-house experts to help maximize results
- Resources fully dedicated to open science

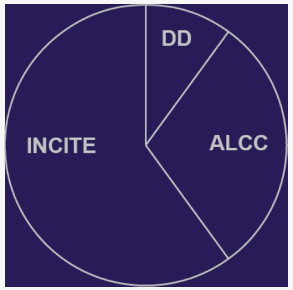




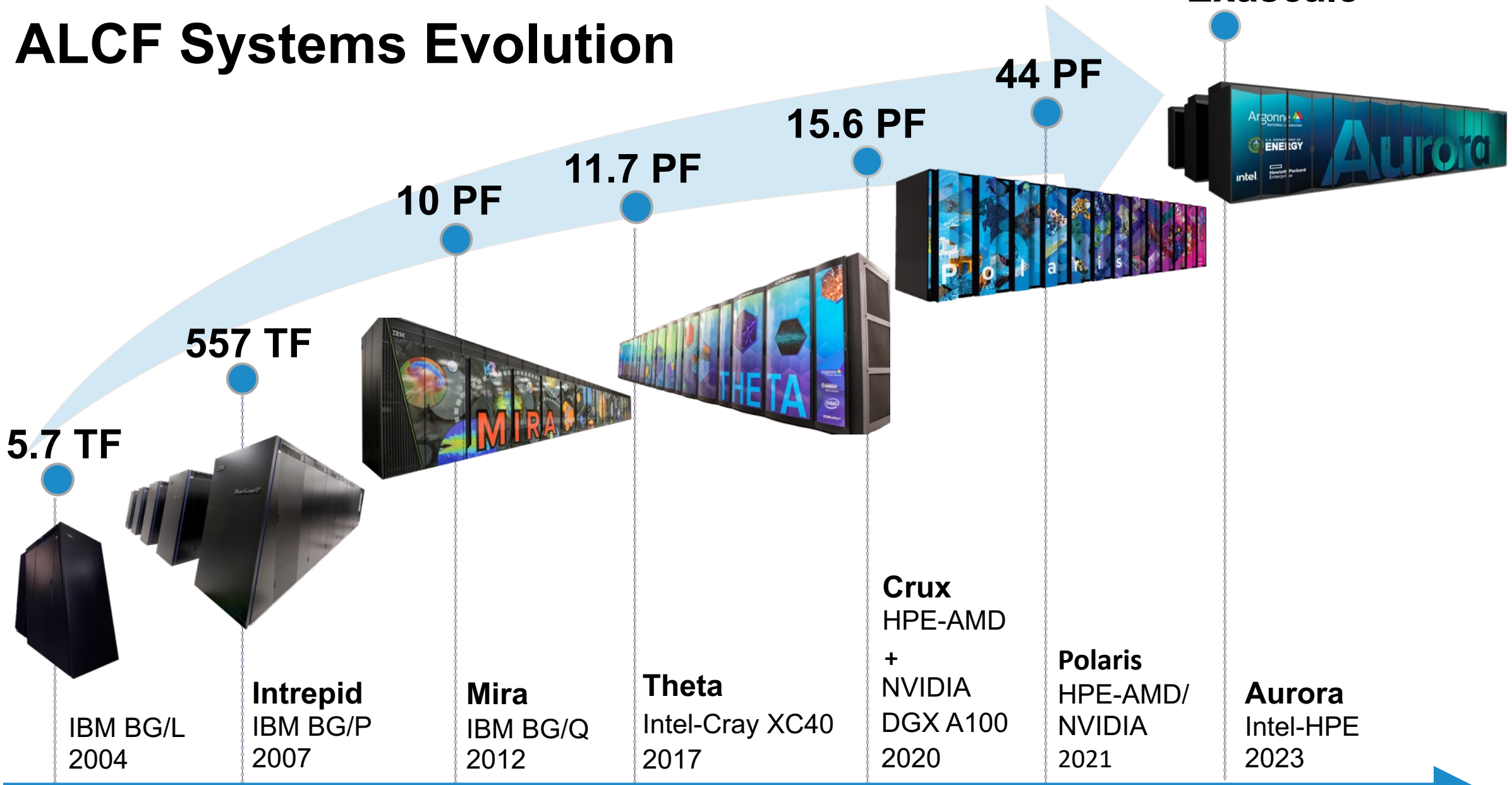
Architecture supports three types of computing

- Large-scale Simulation (PDEs, traditional HPC)
- Data Intensive Applications (scalable science pipelines)
- Deep Learning and Emerging Science AI (training and inferencing)

ALCF offers different pipelines for different applications



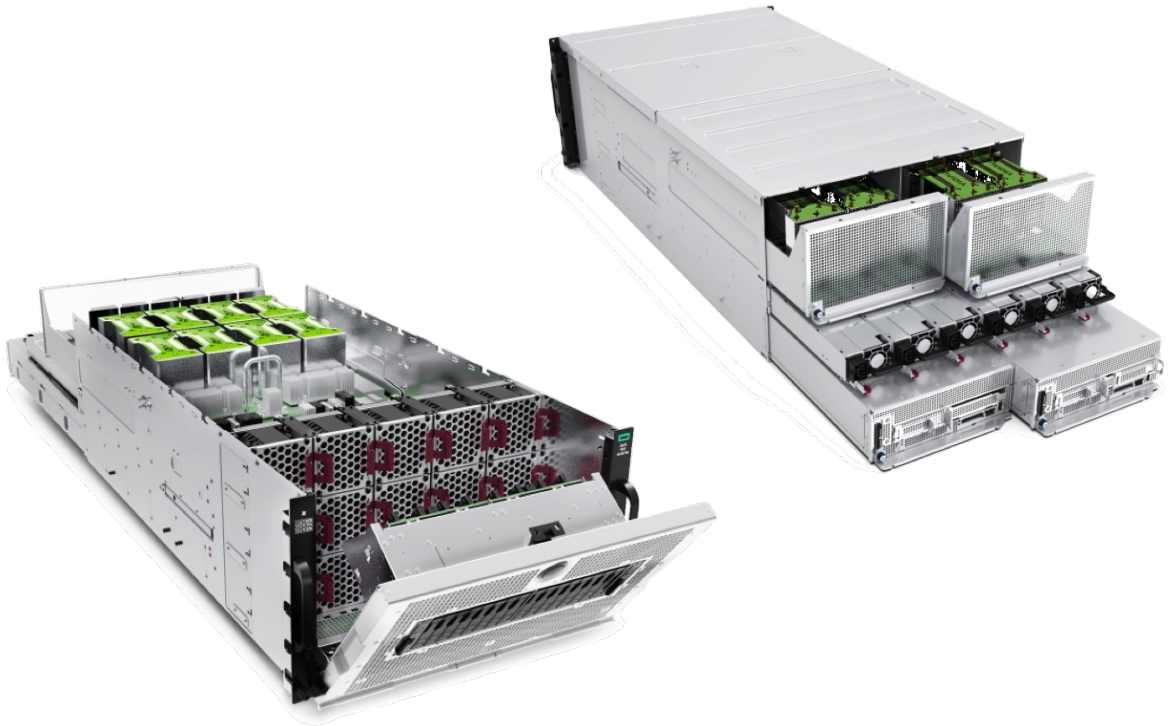
ALCF Systems Evolution



System Overview

Polaris System Configuration

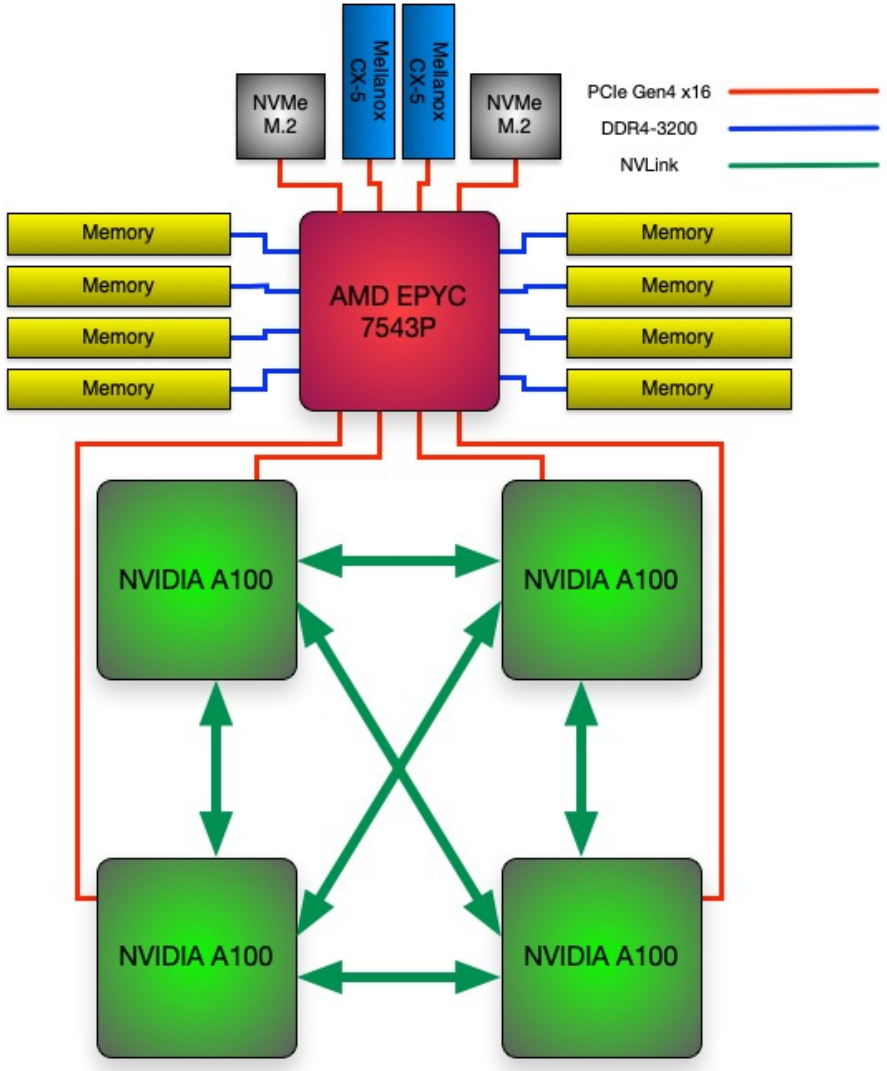
# of River Compute racks	40
# of Apollo Chassis	280
# of Nodes	560
# of AMD EPYC 7543P CPUs	560
# of NVIDIA A100 GPUs	2240
Total GPU HBM2 Memory	87.5TB
Total CPU DDR4 Memory	280 TB
Total NVMe SSD Capacity	1.75 PB
Interconnect	HPE Slingshot
# of Cassini NICs	1120
# of Rosetta Switches	80
Total Injection BW (w/ Cassini)	28 TB/s
Total GPU DP Tensor Core Flops	44 PF
Total Power	1.8 MW



Apollo 6500

Polaris Single Node Configuration

# of AMD EPYC 7543P CPUs	1
# of NVIDIA A100 GPUs	4
Total HBM2 Memory	160 GB
HBM2 Memory BW per GPU	1.6 TB/s
Total DDR4 Memory	512 GB
DDR4 Memory BW	204.8 GB/s
# OF NVMe SSDs	2
Total NVMe SSD Capacity	3.2 TB
# of Cassini NICs	2
Total Injection BW (w/ Cassini)	50 GB/s
PCIe Gen4 BW	64 GB/s
NVLink BW	600 GB/s
Total GPU DP Tensor Core Flops	78 TF



Aurora

Intel® Data Center GPU Max Series

4th Gen Intel XEON Max Series CPU *with High Bandwidth Memory*

Platform
HPE Cray-Ex

Racks - 166
Nodes - 10,624
CPUs - 21,248
GPUs – 63,744

Interconnect
HPE Slingshot 11
Dragonfly topology with adaptive routing
Cassini NIC, 200 Gb/s (25 GB/s), 8 per node
Network Switch:
25.6 Tb/s per switch (64 200 Gb/s ports)
Links with 25 GB/s per direction

Peak FP64 Performance
≧ 2 exaFLOPS

Memory
10.9PiB of DDR @ 5.95 PB/s
1.36PiB of CPU HBM @ 30.5 PB/s
8.16PiB of GPU HBM @ 208.9 PB/s

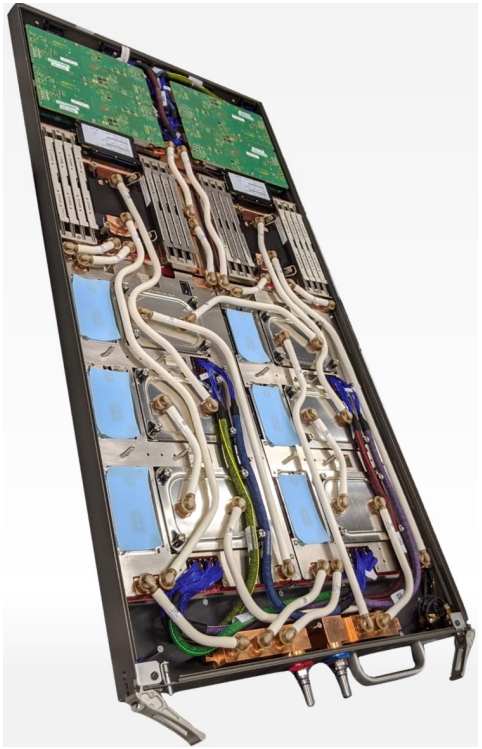
Network
2.12 PB/s Peak Injection BW
0.69 PB/s Peak Bisection BW

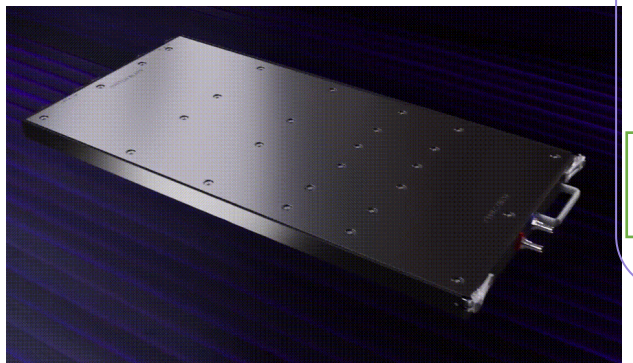
Storage
230PB DAOS Capacity
31 TB/s DAOS Bandwidth

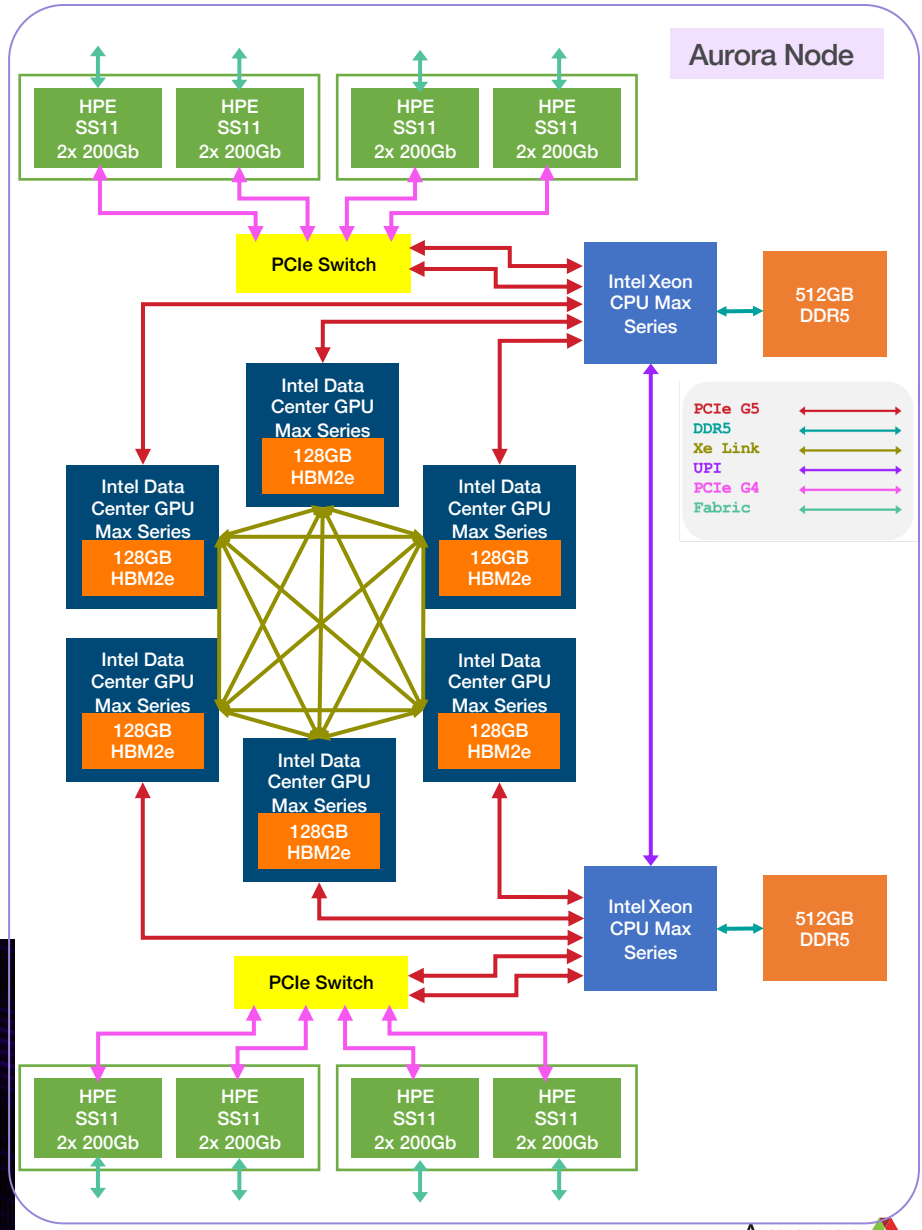
2021	2022	2023	2024
install		early users	production

Aurora Exascale Compute Blade

NODE CHARACTERISTICS
6 GPUs - Intel Data Center GPU Max Series
2 CPUs - Intel Xeon CPU Max Series
768 GB GPU HBM Memory
19.66 TB/s Peak GPU HBM BW
128 GB CPU HBM Memory
2.87 TB/s Peak CPU HBM BW
1024 GB CPU DDR5 Memory
0.56 TB/s Peak CPU DDR5 BW
≧ 130 TF Peak Node DP FLOPS
200 GB/s Max Fabric Injection
8 NICs

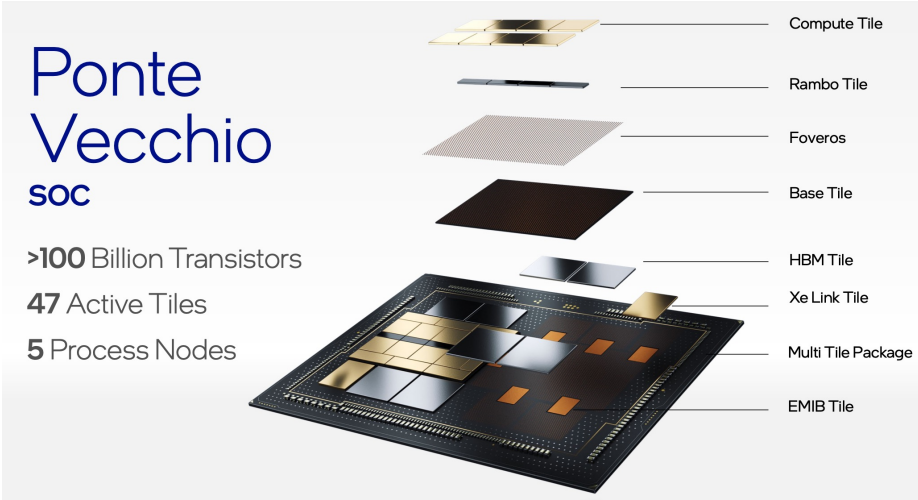






Moving from Polaris to Aurora

- Polaris is based on Nvidia A100s with HPE and Nvidia software.
- New hardware, first large scale discrete GPU from Intel (Based on Intel(R) Data Center GPU Max 1550)
- New software stack to target the new hardware (Intel oneAPI)
- New algorithms and methods in applications, due to push for exascale via the Exascale Computing Project and Argonne's Early Science Project



AURORA | EARLY SCIENCE PROGRAM

ECP
EXASCALE COMPUTING PROJECT

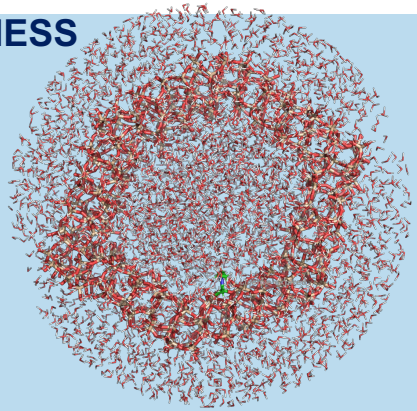
oneAPI™

Applications

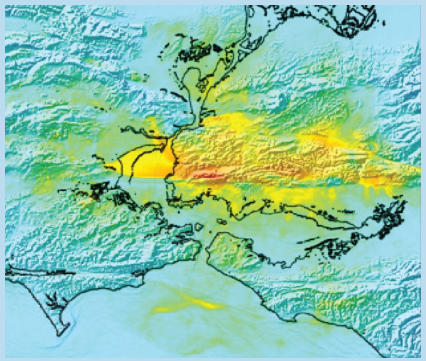
Applications

10 ECP projects
& 4 ESP projects

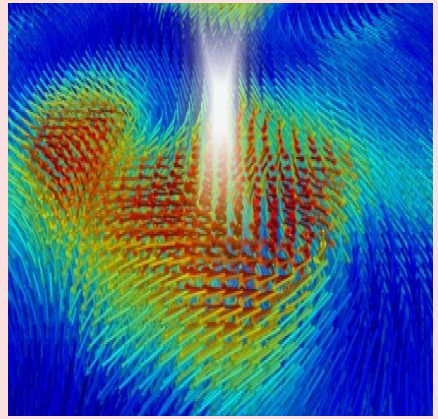
GAMESS



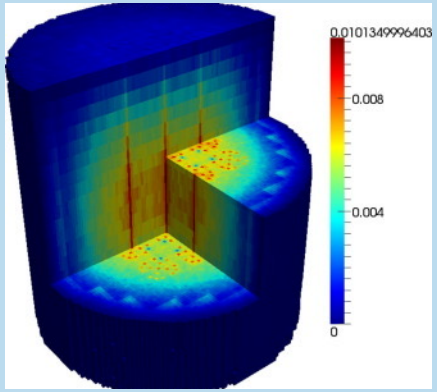
SW4



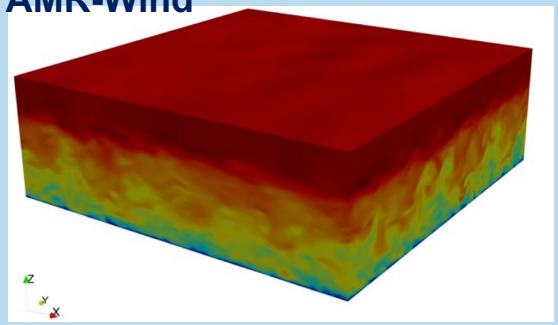
DCMesh



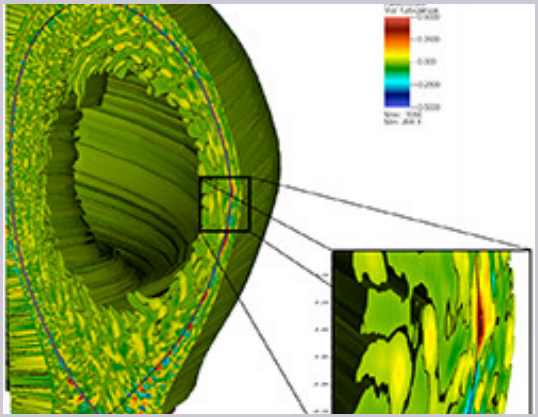
OpenMC



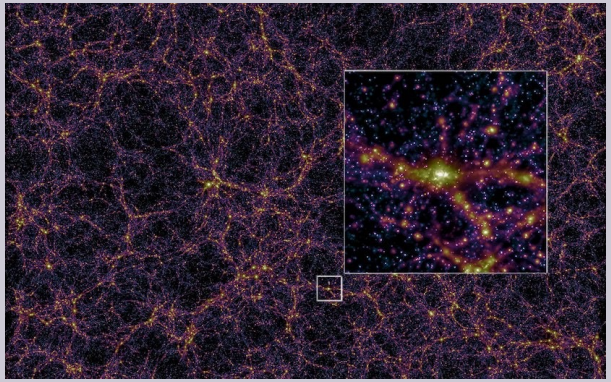
AMR-Wind



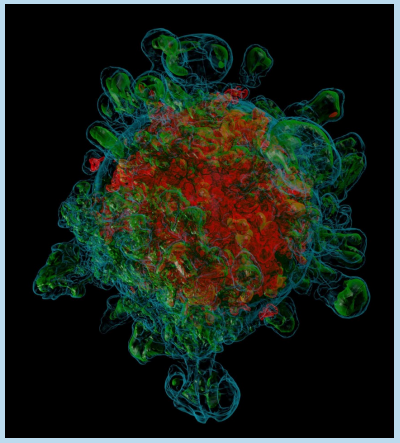
XGC



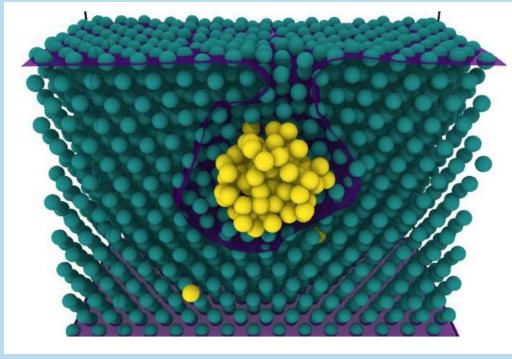
HACC CRK-SPH



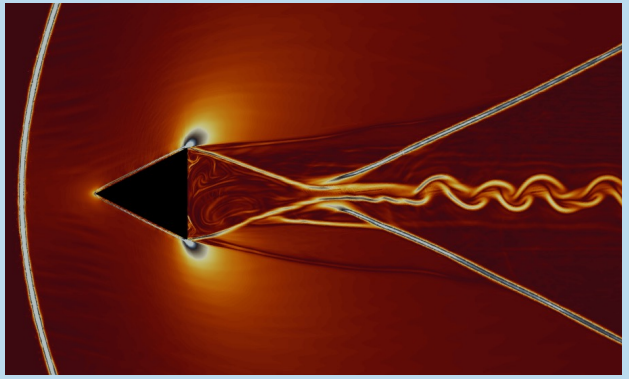
FlashX



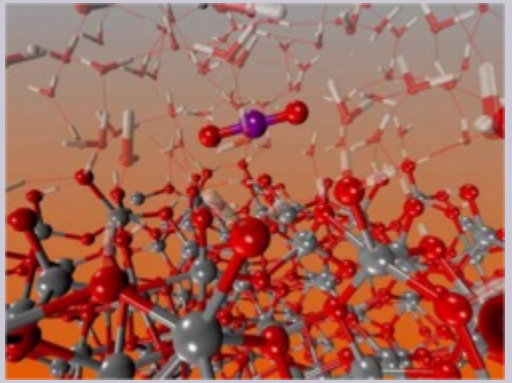
LAMMPS



CNS-libParanumal



NWChemEx



Languages & Programming models/Portability layers

GAMESS
FlashX
DCMesh
OpenMC
HACC CRK-SPH
NWChemEx
XGC
LAMMPS
SW4
CNS-libParanumal
AMR-Wind

OpenMP
OpenACC
SYCL™
RAJA
AMReX

Transitioning Applications to Aurora: Tracking Application and Pre-Production Software Progress on Aurora

Moving to Aurora

- Challenging with new hardware, software, and newly-developed applications for exascale

- How to help ease the transition for application developers and ensure that Aurora is a useful system for applications?

  —Tracking application implementation and porting progress to help identify wide-spread issues early on

  —Tracking application developer-reported bugs in an ease-to-add-to repository that was shared with Intel

- At this point in the preparations, how are applications performing now?

  —Compare performance for eleven of the tracked applications on Aurora and Polaris

Tracking Application Progress

- Over forty applications from the Argonne Early Science Program (ESP) and the Exascale Computing Project (ECP).

- The Argonne Early Science Program (ESP) : 19 projects
  - 9 Simulation projects
  - 5 Learning projects
  - 5 Data projects
- The US DOE Exascale Computing Project (ECP) : 15 projects
  - Some projects contain multiple codes

- Application preparation for Aurora consisted of:
  - Implementation of new algorithms and science
  - Porting to a programming model available on Aurora
  - Identifying and resolving issues
  - Tuning and scaling across the Aurora system

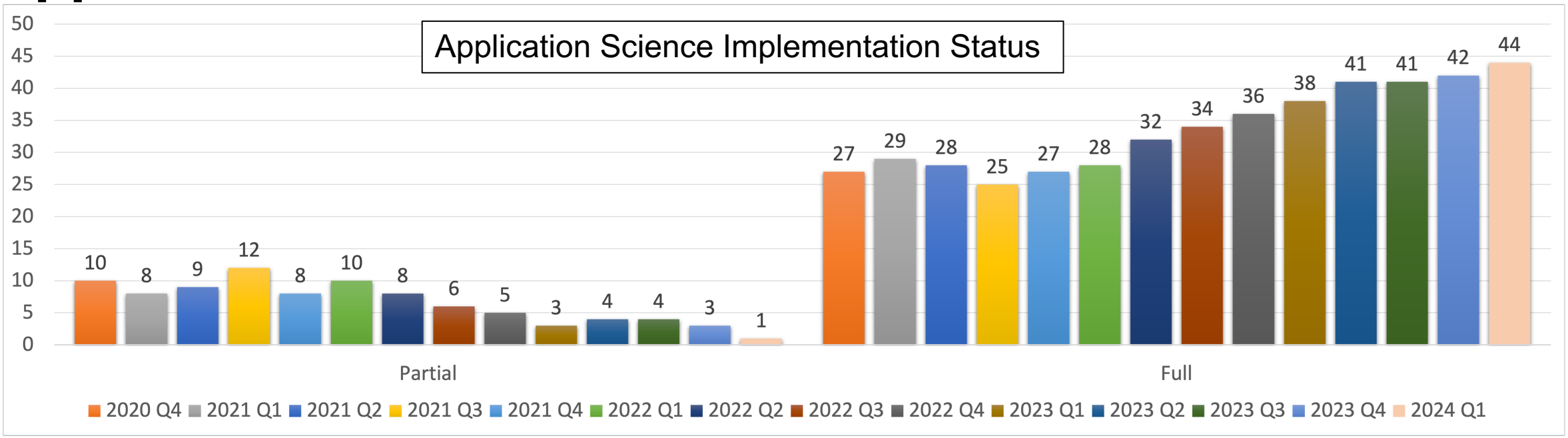
Tracking Application Progress

- To help ease the transition to Aurora, a multi-year effort was undertaken to prepare the applications
- Aurora Applications Working Group
  - —Worked closely with vendor and application developers for many years
- Aurora application development teams were surveyed to quantify the status of their efforts and understand any widespread issues
- What are the results?

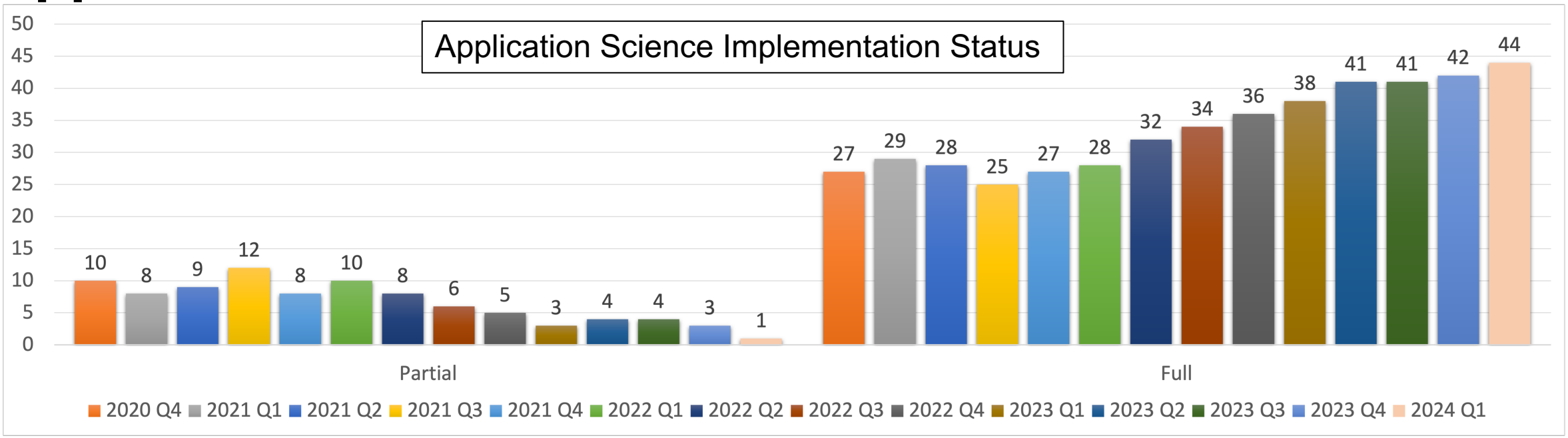
Application Name:
Project Name(s):
Project Type (ESP/ECP):
Project PI(s):
ANL Contact(s):
Intel Contact(s):
People Needing JLSE Access:
Programming Languages:
Programming Models:
Major external dependencies:
Code Intended to Run on PVC?
Code Actively Engaged?
Status of Science/Algorithmic Development
Explanation of Science/Algorithm State
Work Remaining for Codes not "Fully Implemented"
Status implementation with an Aurora Programming Model
Explanation of Porting State
Work Remaining for Codes not "Full Ported"
Components Running on PVC
Reasons Not Running on PVC
All Significant Issues Listed
Information or Support Needed from Intel
Able to Run on Aurora?
Work Need to Run on Aurora
Confidence Code Will be Ready by end of 2023

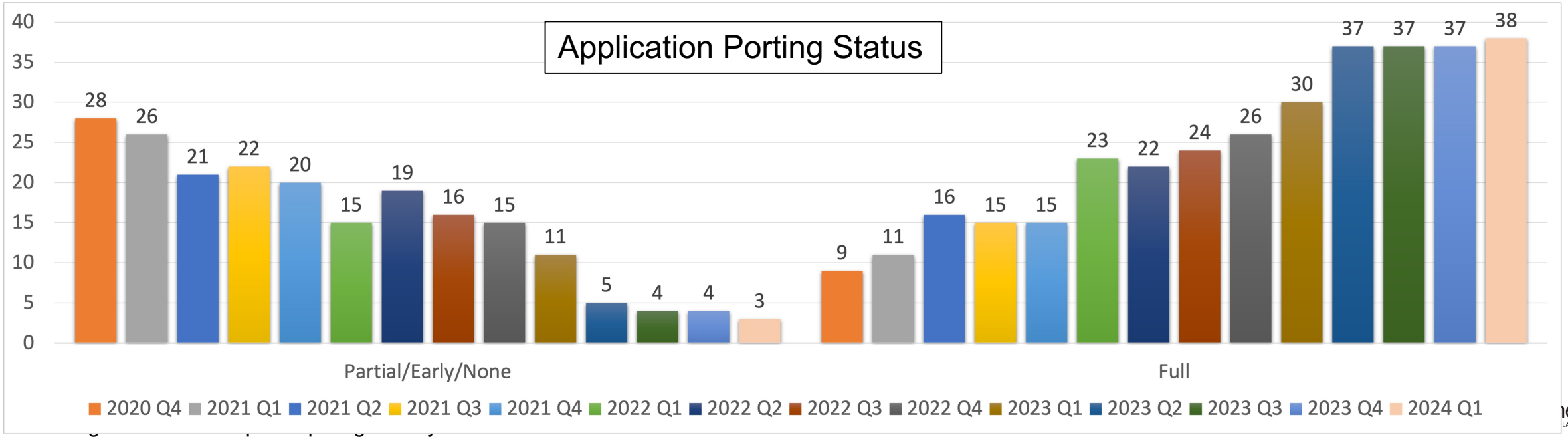
Other Issues
Highlights
Works with VTune?
Explanation of VTune Problems
Plan for Multi-tile & Multi-GPU
In Sunspot Test Suite
FOM Description
Code Version for FOM
Input Deck for FOM
FOM – Sunspot or Aurora Single GPU
Tiles used for Sunspot Single GPU FOM
FOM - NVIDIA A100
FOM - AMD MI-250
FOM - CPU
Performance Comments
Scale of runs on Sunspot
Sunspot Comments
Sunspot Scaling Plot
Scale of runs on Aurora
Aurora Comments
Aurora Scaling Plot

Application Aurora Status

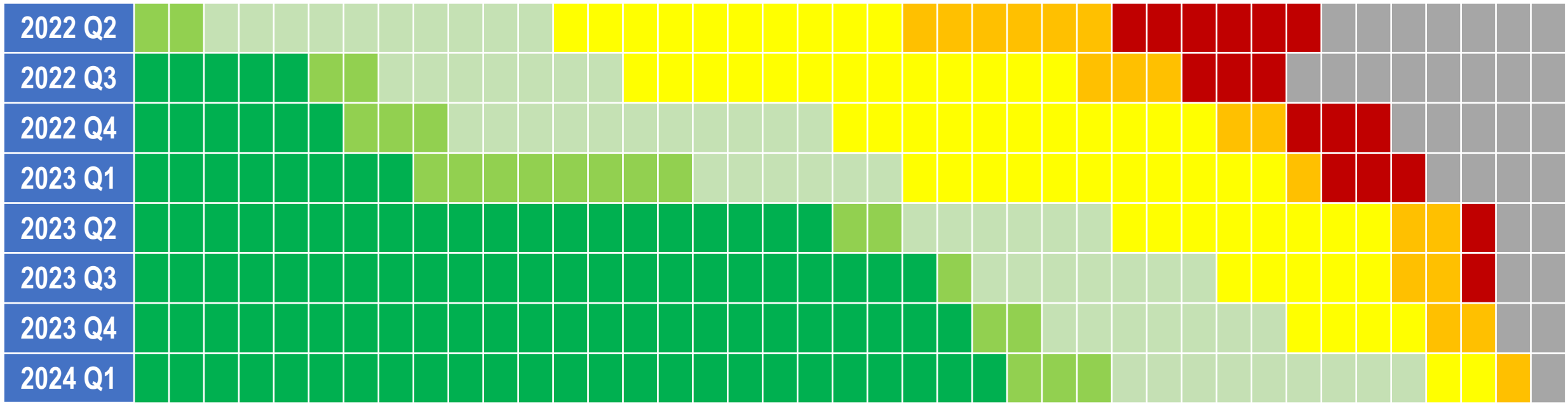


Application Aurora Status





Application Aurora Running Status

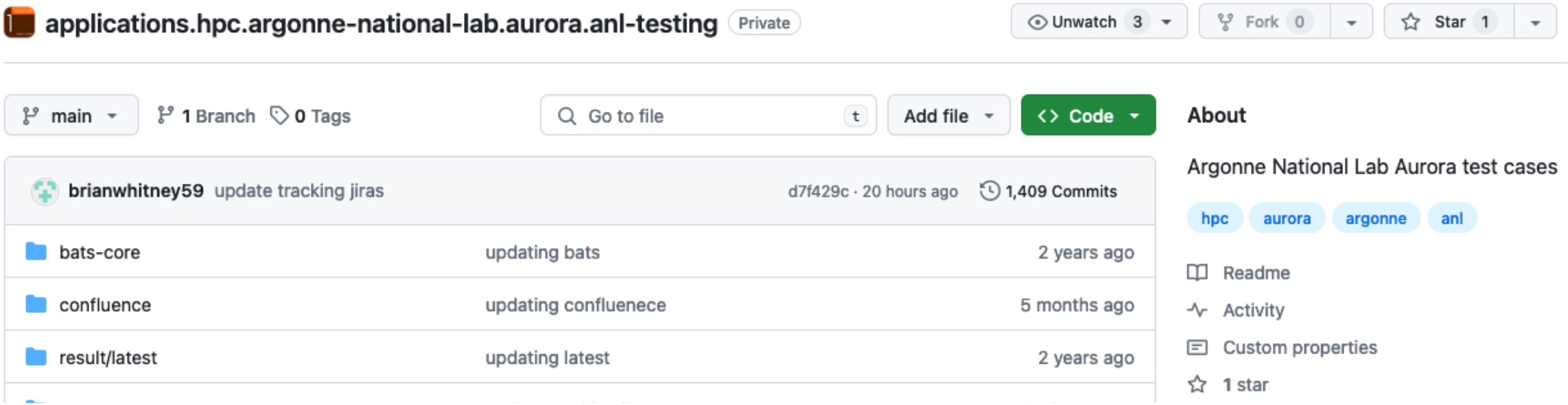


Over multiple years, with a lot of effort, most of the tracked applications have their science implemented, are ported to Aurora, and are running well

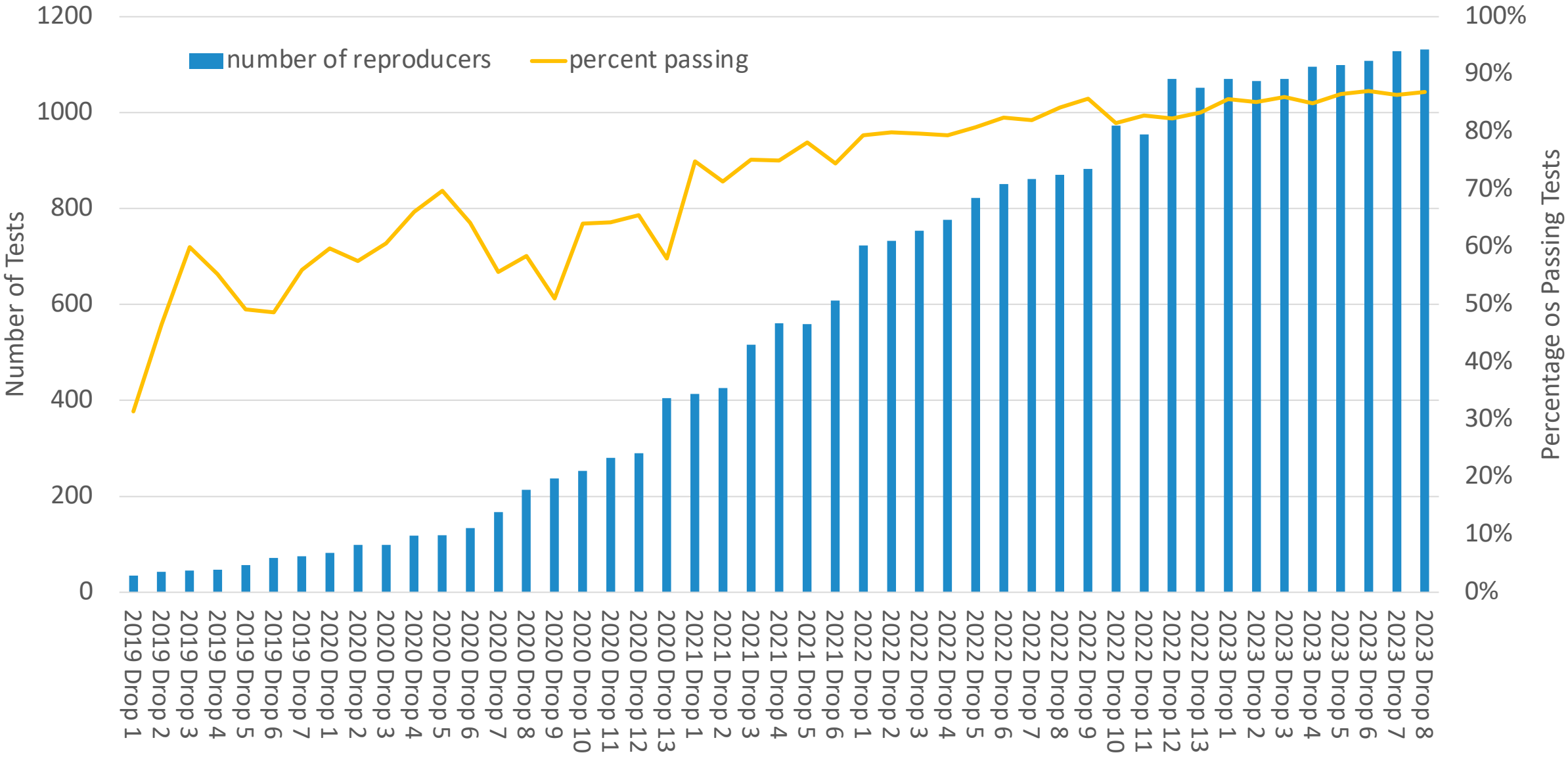
[0] https://github.com/bats-core/bats-core/

Tracking Software/Bug/Issue Progress

- To facilitate the identification and resolution of issues in the Aurora software stack, Argonne staff constructed a bash-based test set and framework based on bats-core [0]
- The test set and framework is in a private git repository which is shared with Intel
- Our motivation to use bats-core was to make it very simple to add bug reproducers
- The testing framework began with only a few initial tests and grew to contain over 1100 tests
- Facilitated communication and discussions about bugs with the vendor
- Useful tool for validating systemic underlying changes in the environment



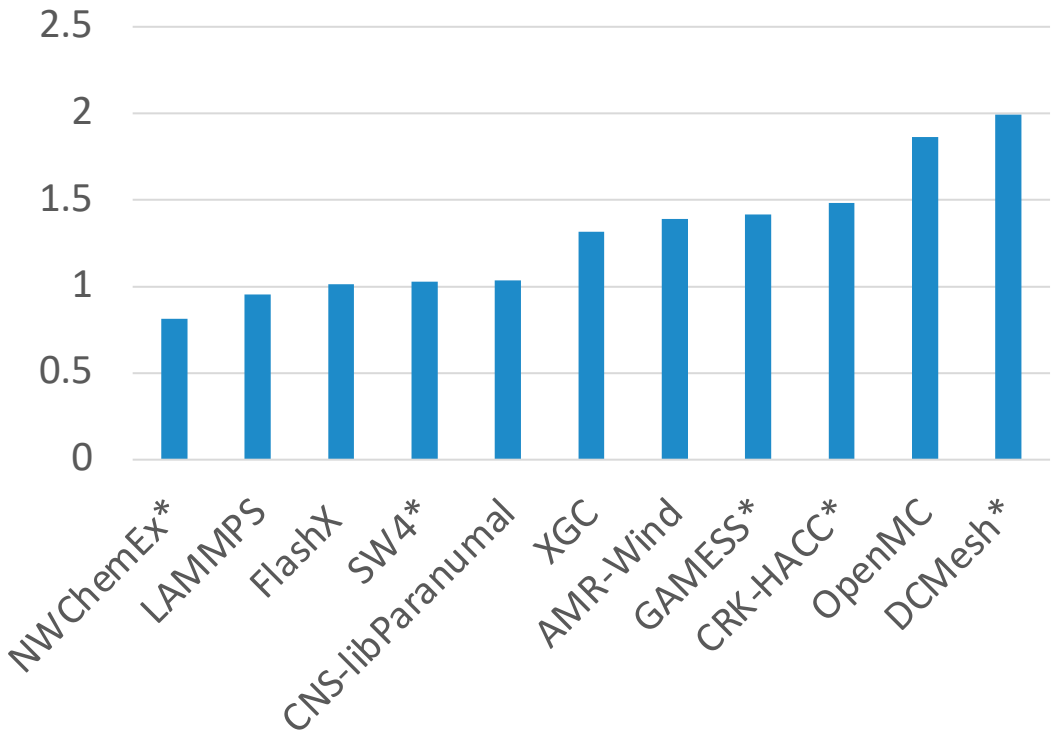
Reproducers In the Test Set Over Time (2019 to 2023)



Over multiple years, with a lot of effort, most of the main issues are resolved

Performance Comparison for Aurora and Polaris

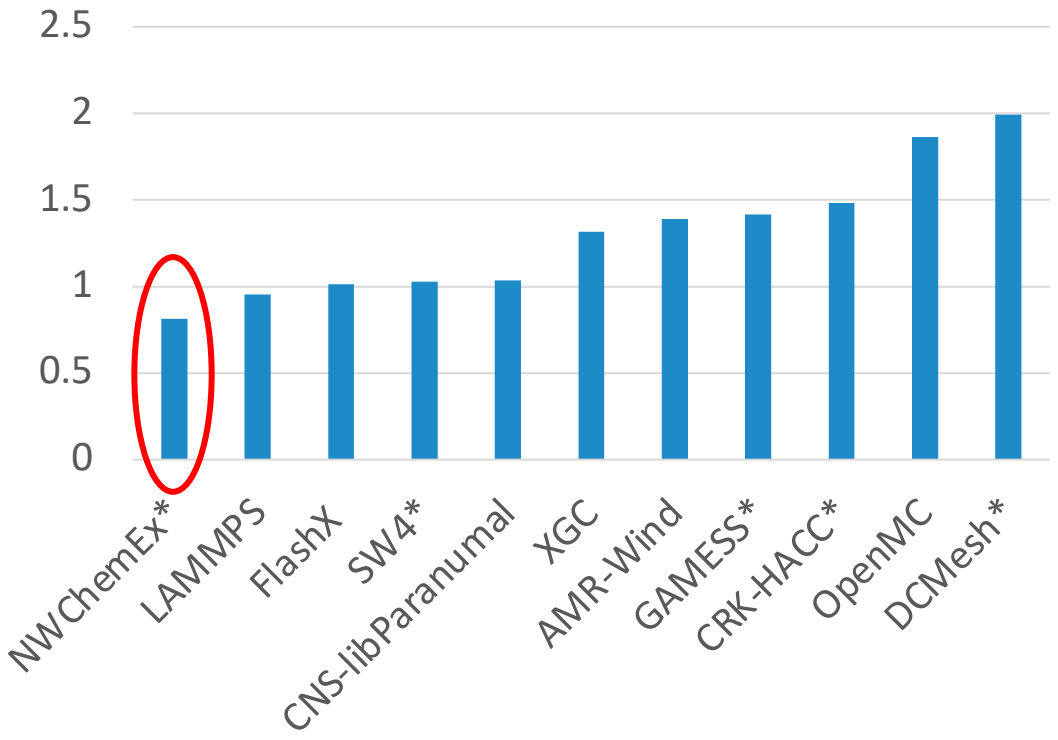
FOM Performance on a PVC, Relative to A100



*Result from Sunspot

> 1 means FOM performance is better on Aurora, < 1 means FOM performance is better on Polaris

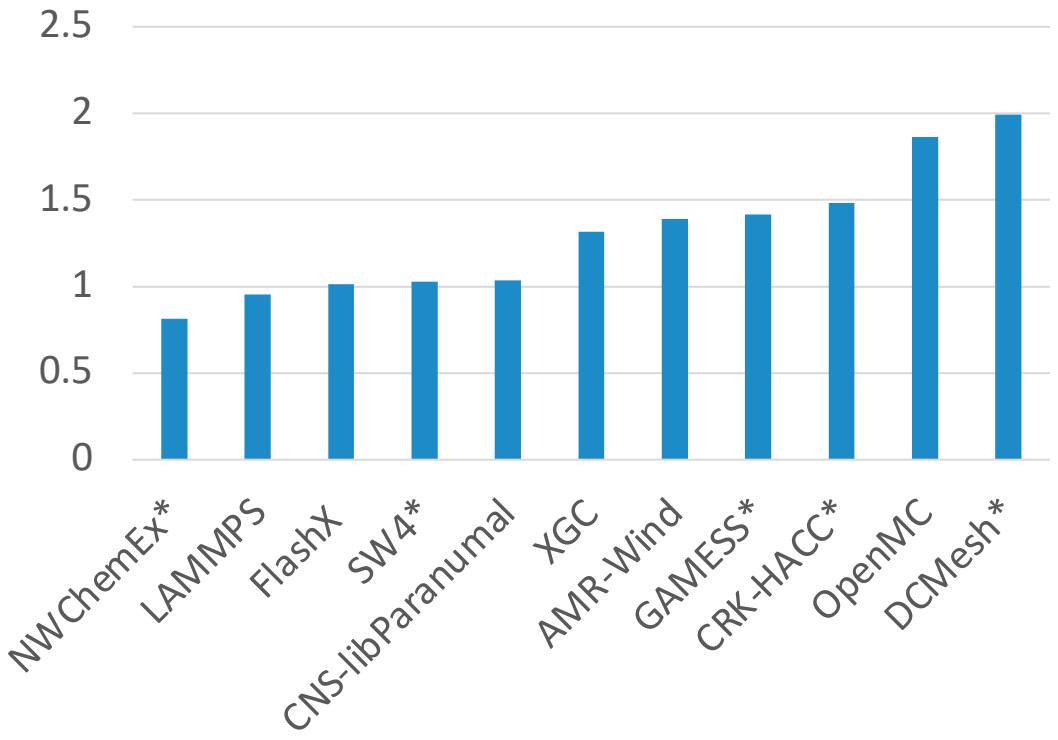
FOM Performance on a PVC, Relative to A100



*Result from Sunspot

Due to a limitation of the version of SYCL on Aurora which results in needing more synchronization on Aurora than on Polaris

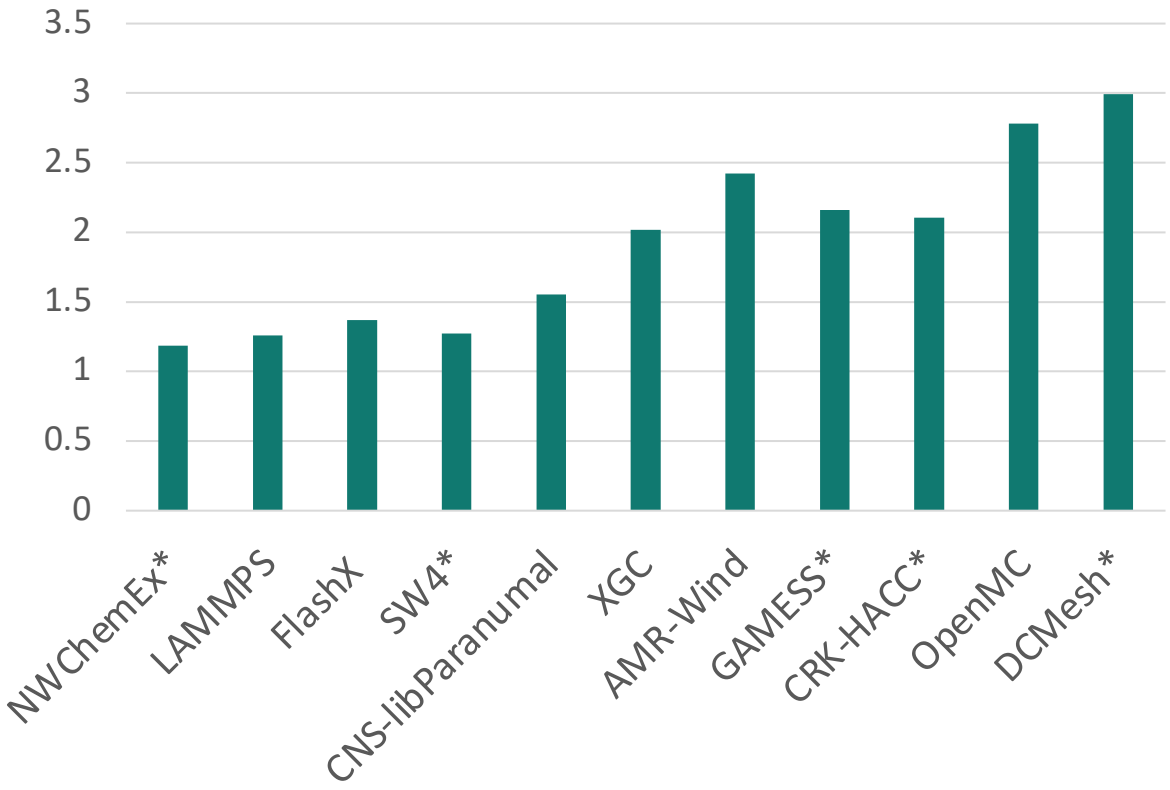
FOM Performance on a PVC, Relative to A100



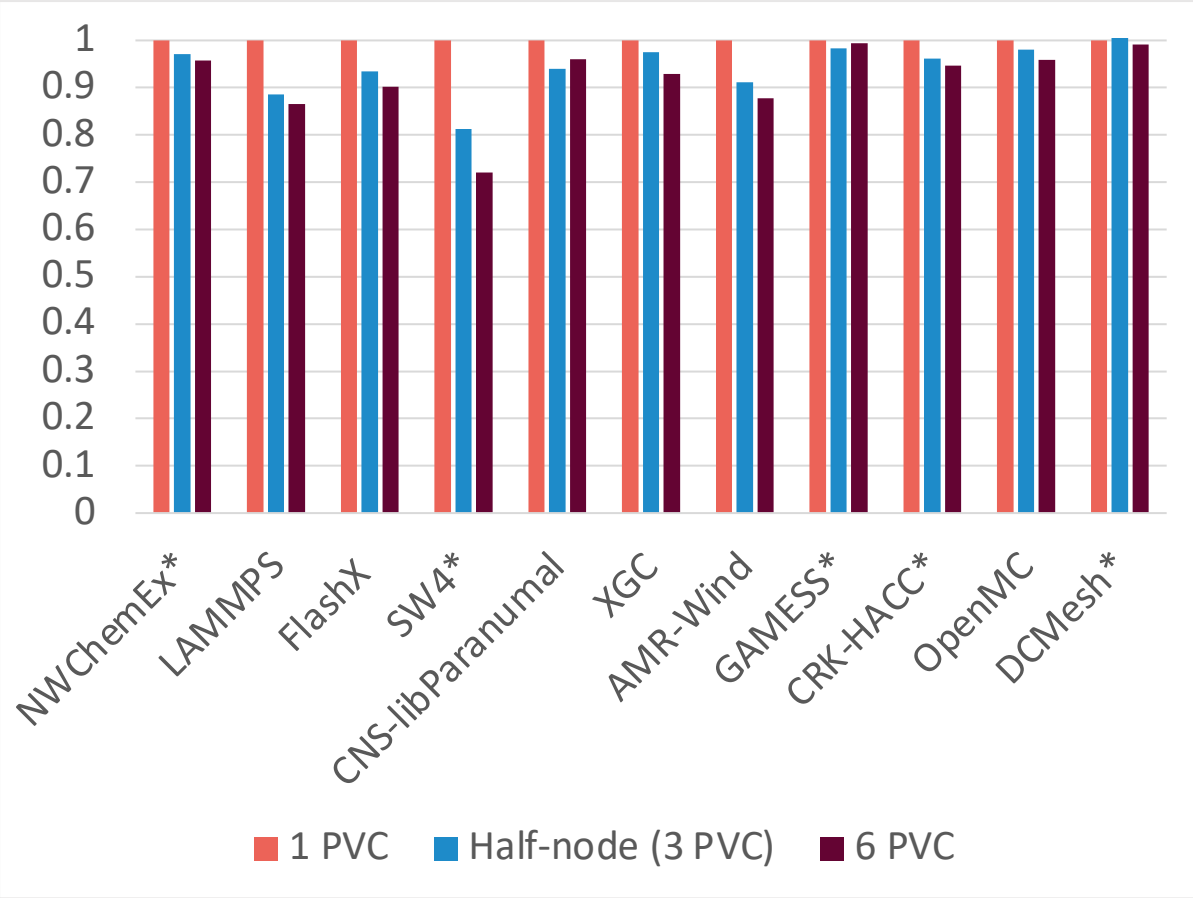
*Result from Sunspot

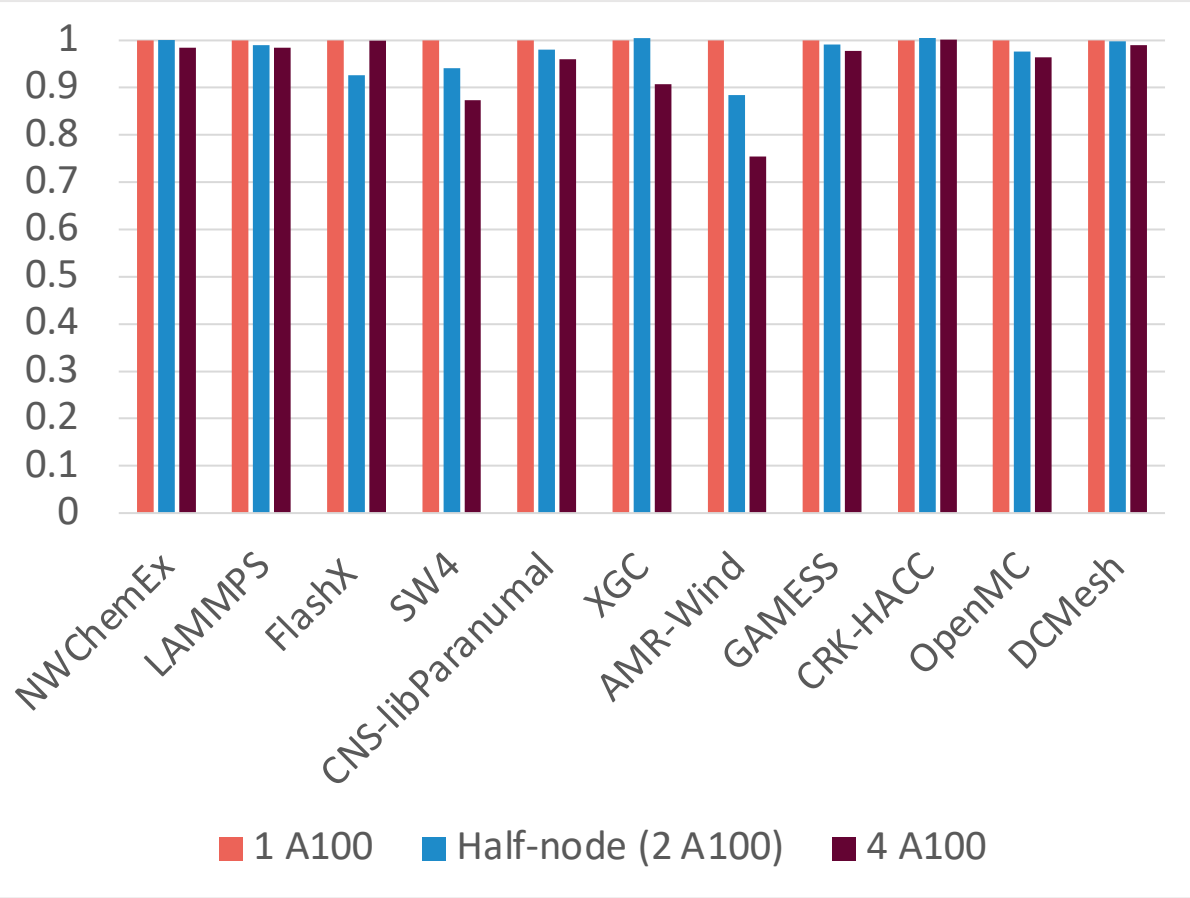
> 1 means FOM performance is better on Aurora, < 1 means FOM performance is better on Polaris

FOM Performance on a Aurora node, Relative to a Polaris node



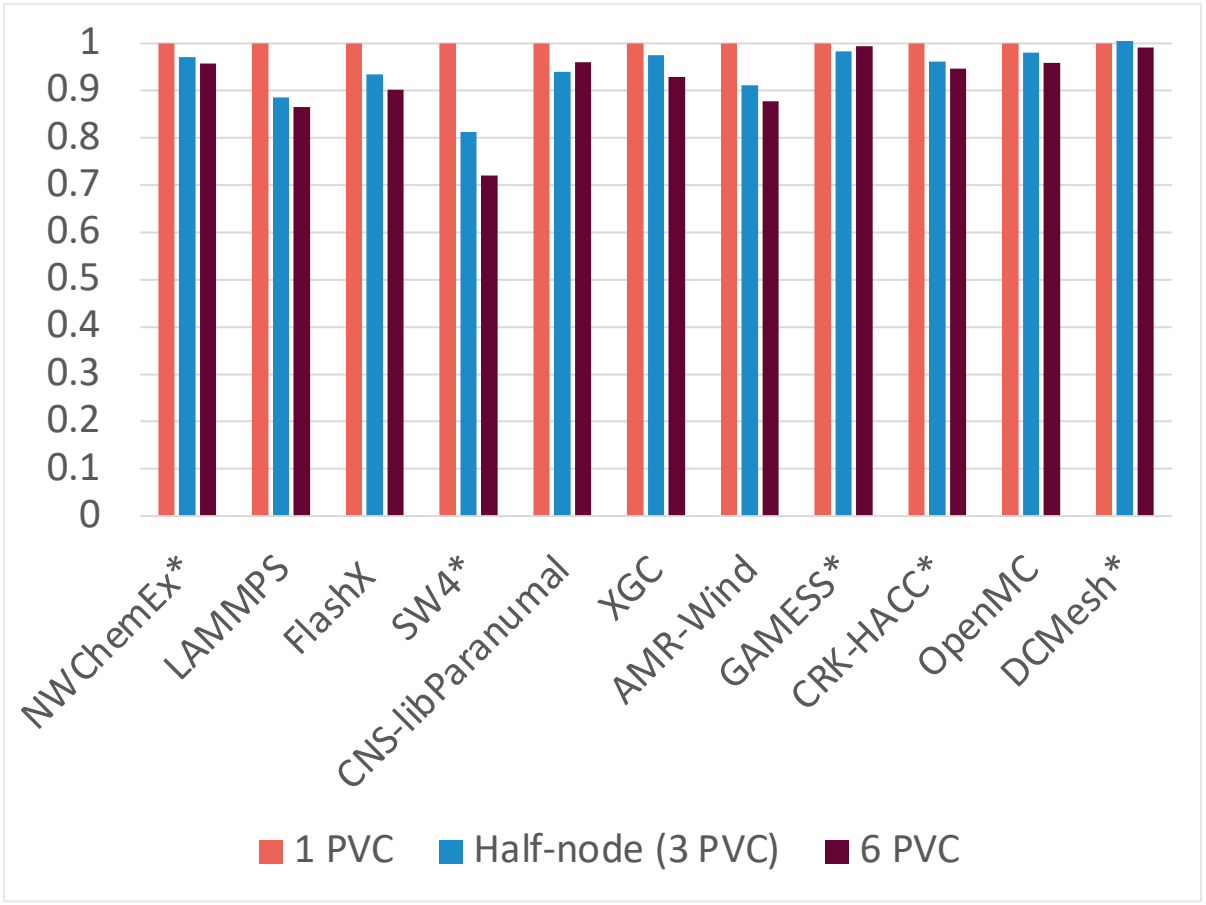
Intra-Node Parallel Efficiency

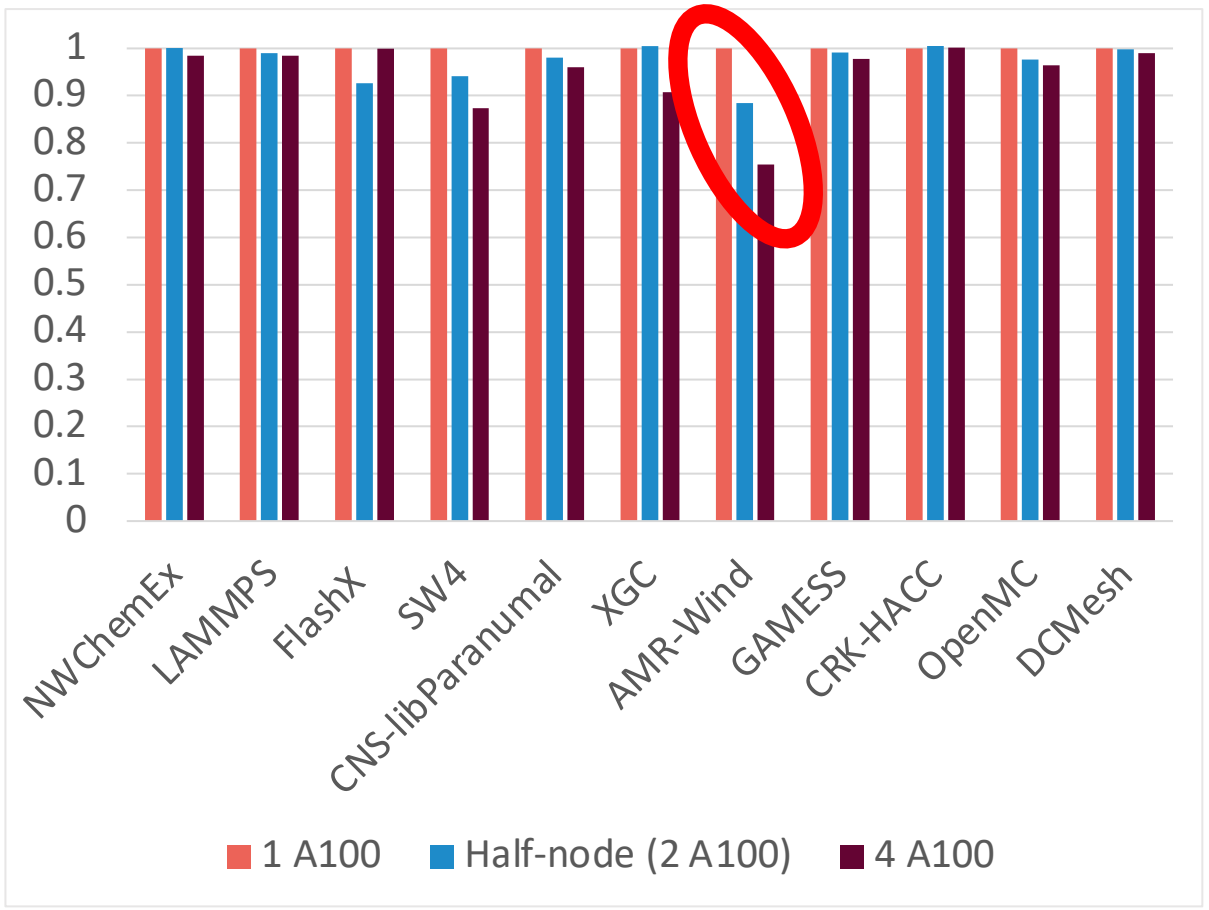




*Result from Sunspot

Intra-Node Parallel Efficiency

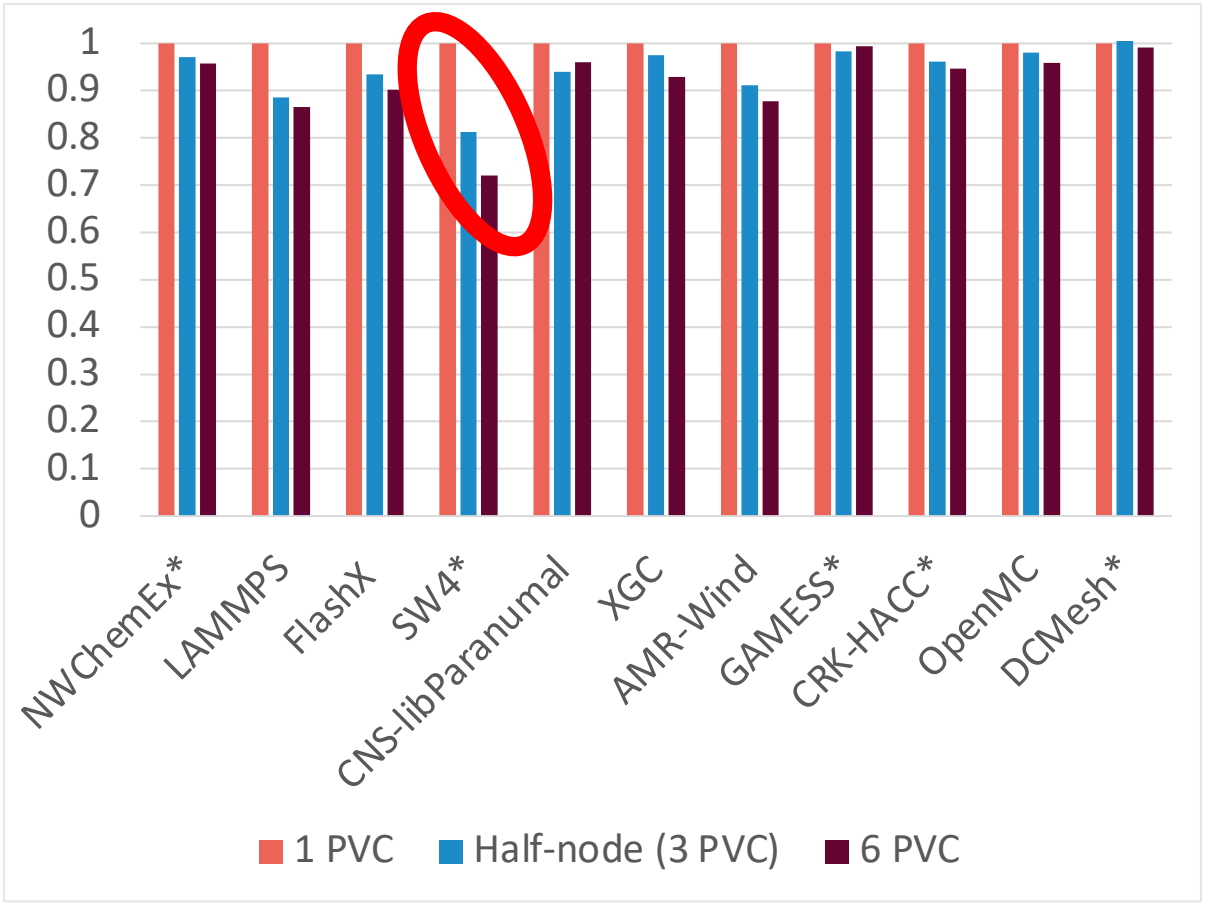


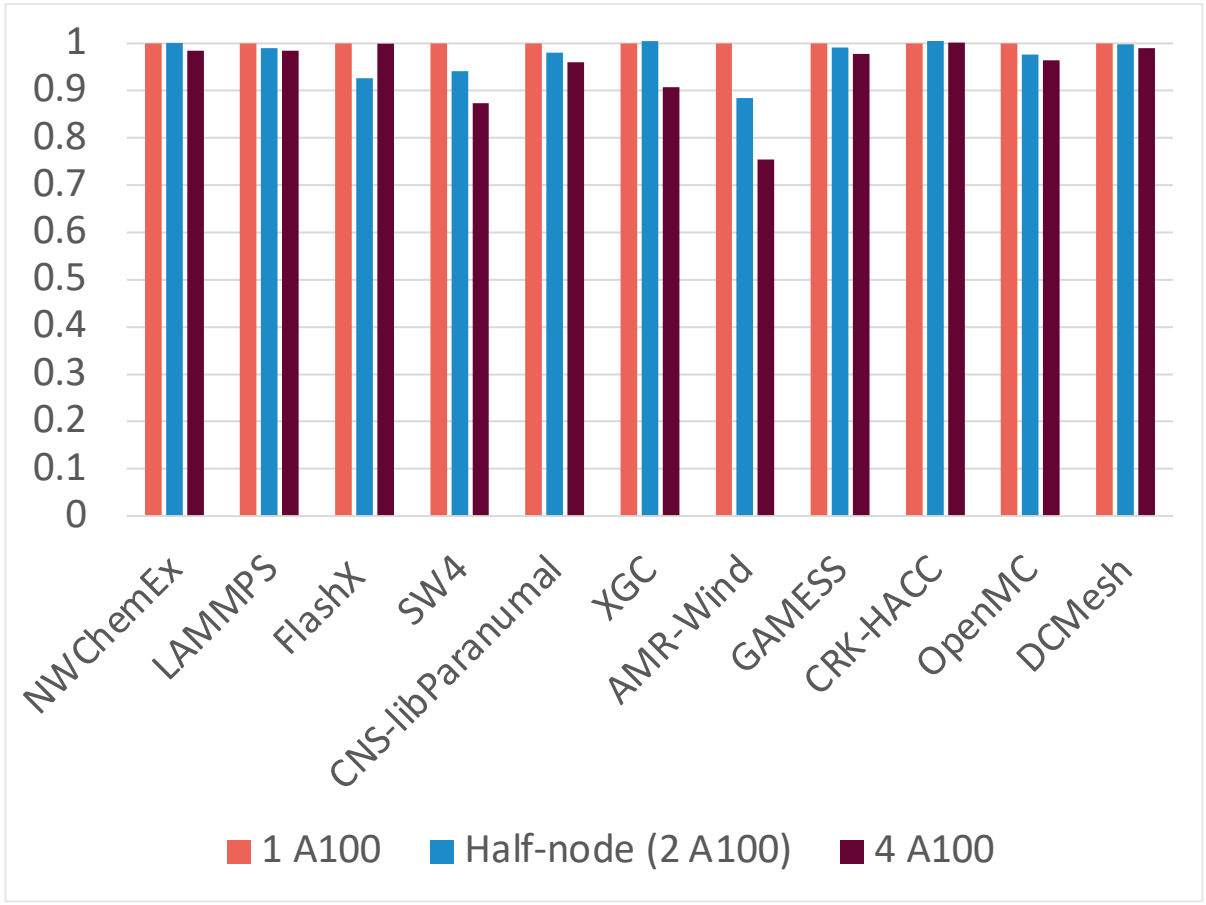


*Result from Sunspot

- AMR-Wind is weak-scaling
- Aurora PCIe bandwidth for transferring data between CPU and GPU increased linearly with more GPUs, while on Polaris, bandwidth increased less than linearly
- Further investigation needed

Intra-Node Parallel Efficiency





*Result from Sunspot

- SW4 is strong-scaling
- From profiling, additional communication between the 12 MPI ranks per node on Aurora compared to the 4 ranks per node on Polaris.
- Further investigation needed

Observations from Porting to Aurora

Observations from Porting to Aurora

- Several applications already had OpenMP offload or CUDA applications to start from
  - Two of the applications used SYCLomatic to port to SYCL from CUDA

- Common ways to improve performance
  - Avoid register spilling
    - Decreasing SIMD size and enabling a larger register file per thread
  - Tuning the workgroup size for different kernels

- Issues with C++ feature support in device code

- Still some issues!
  - A current issue related to SYCL's host_task functionality limits the performance on Intel GPUs. As a work-around, the application enforces a synchronization of the SYCL event returned by the host_task for the Level-Zero plugin associated with Intel GPUs. (NWChemEx)

Concluding Remarks and Takeaways

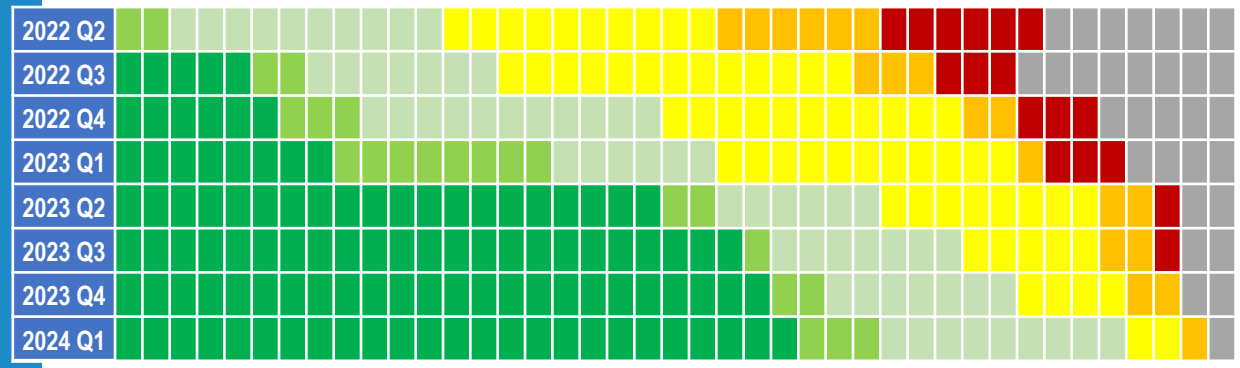
Concluding Remarks and Takeaways

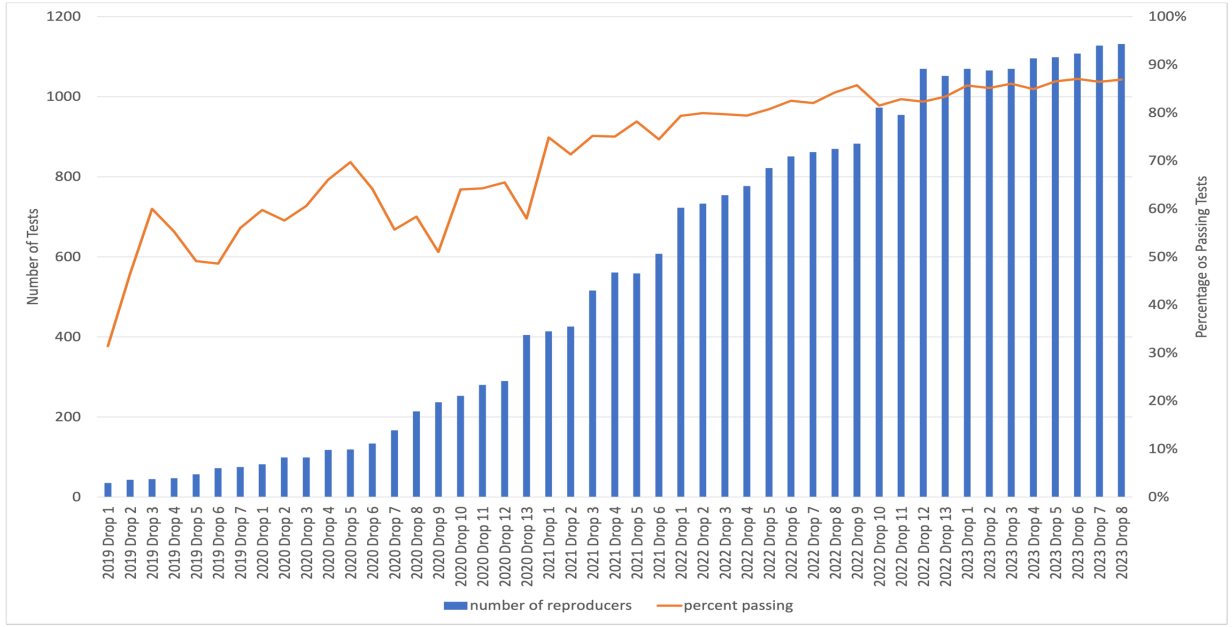
1. Our goal is to ensure applications will run well on Aurora and are transitioning from Polaris well
    - This is complicated since the hardware and software were being developed by the vendor at the same time as application developers were adding new methods to target exascale

Concluding Remarks and Takeaways

1. Our goal is to ensure applications will run well on Aurora and are transitioning from Polaris well
   - This is complicated since the hardware and software were being developed by the vendor at the same time as application developers were adding new methods to target exascale

2. To help with this, we worked closely with vendors and over forty applications for multiple years
   - Tracked application and software/bug reproducers, encountered and resolved many issues
   - As of 2024, almost all tracked applications are running well on Aurora, and most of the bug reproducers are passing



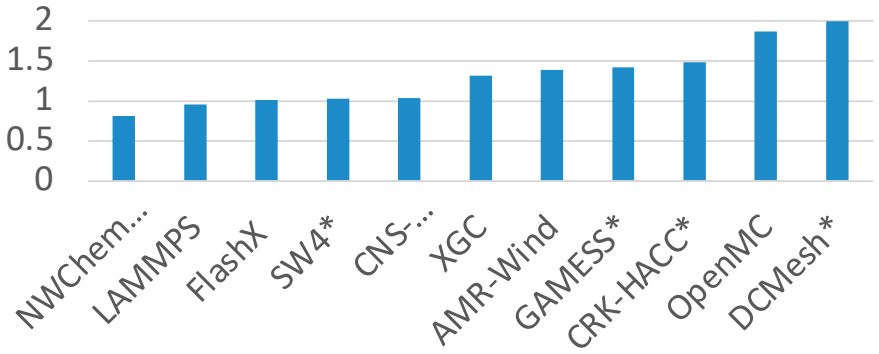


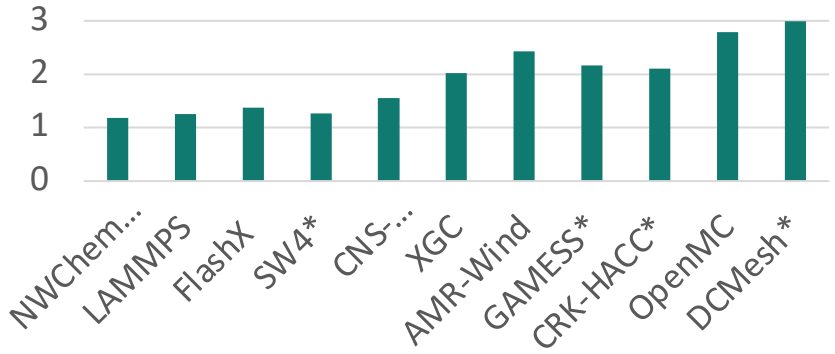
Concluding Remarks and Takeaways

1. Our goal is to ensure applications will run well on Aurora and are transitioning from Polaris well
   - This is complicated since the hardware and software were being developed by the vendor at the same time as application developers were adding new methods to target exascale

2. To help with this, we worked closely with vendors and over forty applications for multiple years
   - Tracked application and software/bug reproducers, encountered and resolved many issues
   - As of 2024, almost all tracked applications are running well on Aurora, and most of the bug reproducers are passing

3. Performance comparisons of eleven of the tracked applications from a wide variety of science domains and programming models
   - On average the FOM performance was 1.3x greater on a single GPU of Aurora than on Polaris and 1.9x greater on a single node of Aurora than on Polaris
   - The intra-node parallel efficiency of the set of applications was similar between Aurora and Polaris





Concluding Remarks and Takeaways

1. Our goal is to ensure applications will run well on Aurora and are transitioning from Polaris well
    - This is complicated since the hardware and software were being developed by the vendor at the same time as application developers were adding new methods to target exascale

2. To help with this, we worked closely with vendors and over forty applications for multiple years
    - Tracked application and software/bug reproducers, encountered and resolved many issues
    - As of 2024, almost all tracked applications are running well on Aurora, and most of the bug reproducers are passing

3. Performance comparisons of eleven of the tracked applications from a wide variety of science domains and programming models
    - On average the FOM performance was 1.3x greater on a single GPU of Aurora than on Polaris and 1.9x greater on a single node of Aurora than on Polaris
    - The intra-node parallel efficiency of the set of applications was similar between Aurora and Polaris

4. Aurora is still in pre-production and we're continuing to work to improve performance of applications.
5. Aurora is shaping up to be a good platform for applications!

Acknowledgements

- This work was done on a pre-production supercomputer with early versions of the Aurora software development kit.
- This work was supported by the Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC02-06CH11357, and by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of two U.S. Department of Energy organizations (Office of Science and the National Nuclear Security Administration).
- We would like to thank our collaborators from **Programming Models and Architectures (PMA)** working group at Argonne National Laboratory: Kevin Harms, Vitali Morozov, Kris Rowe, Longfei Gao, and Brice Videau.

Thank you!

Backup