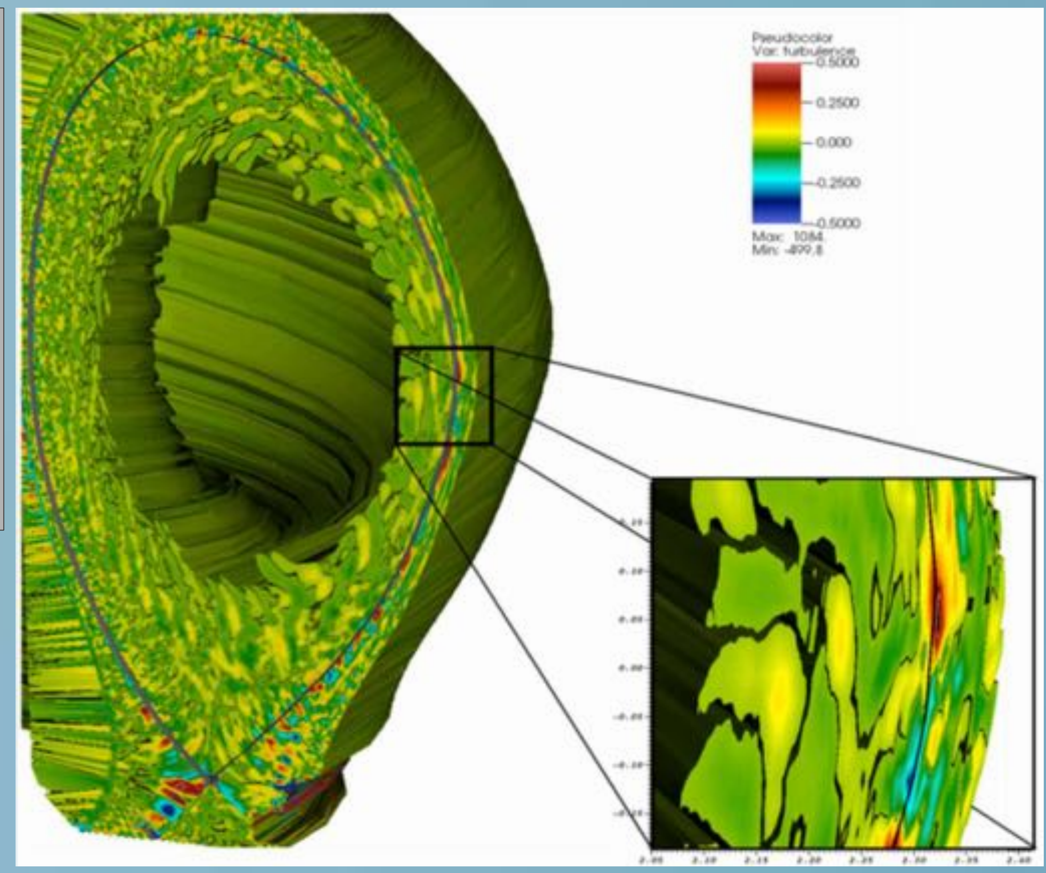
Data Management Challenges In HBPS

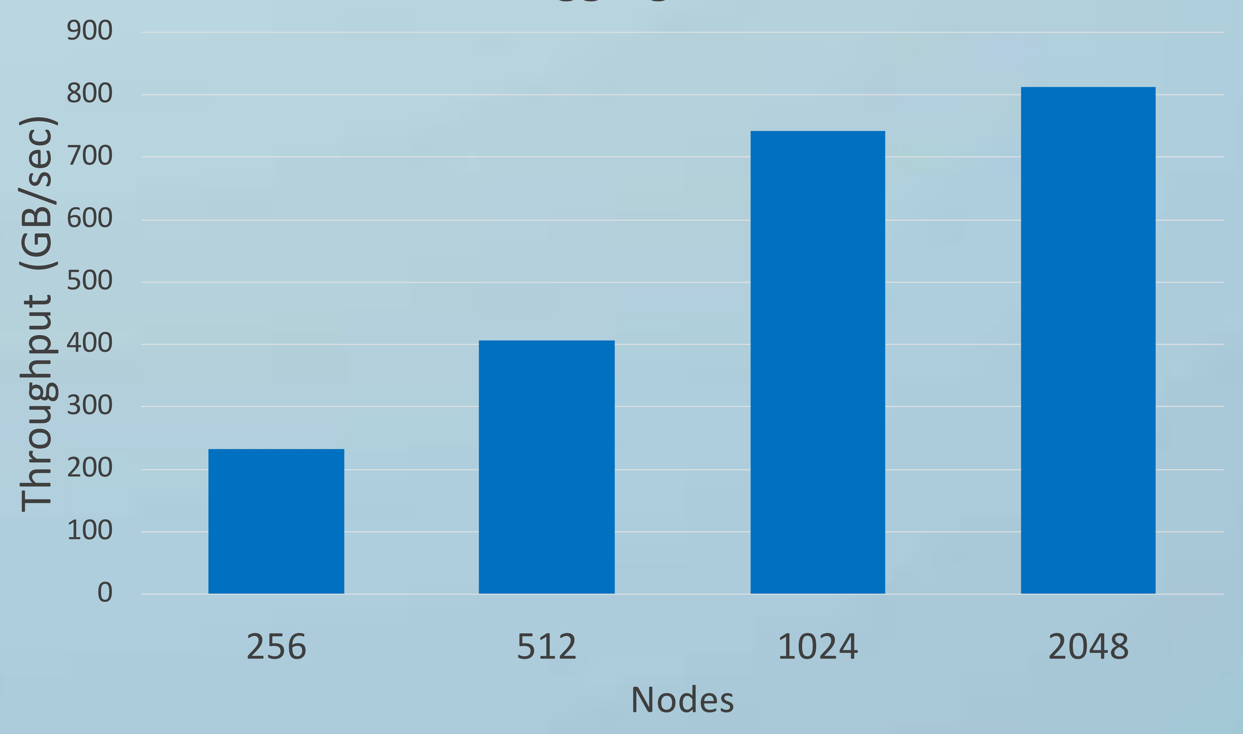
HBPS High-fidelity Boundary Plasma Simulation

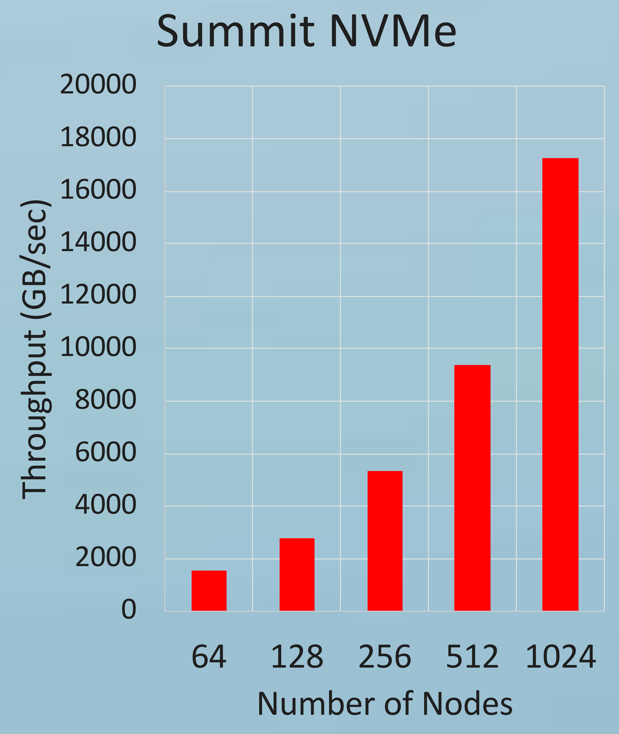
Jong Youl Choi[1], Michael Churchill[2], Davide Curreli[3], Sonata Mae Valaitis[3], Robert Hager[2], Seung-Hoe Ku[2], E. D'Azevedo[1], Bill Hoffman[4], David Pugmire[1], Scott Klasky[1], C. S. Chang[3]
[1]ORNL, [2]PPPL, [3]Univ. of Illinois Urbana-Champaign, [4]Kitware

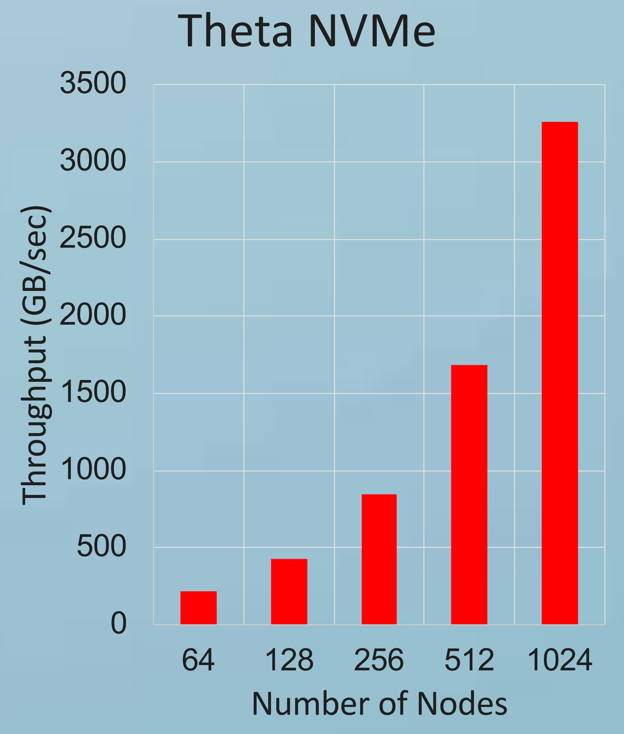


XGC I/O Performance

We maintain cutting edge I/O performance for XGC on various file systems, including SSDs and NVMe, on Cori, Theta, and Summit.







Our team continues to innovate to take full advantage of the new memory and storage technologies, and to provide the highest levels of performance.

I/O System	Summit ORNL	Theta ANL	Cori NERSC
Locality	Node local	Node local	Remote Shared
System	Local filesystem	Local filesystem	Cray WARP
Capacity	800 GB per node	128 GB per node	288 Server 50 TB limit per job
Parallel Filesystem	GPFS Lustre	Lustre	Lustre

XGC Software Process

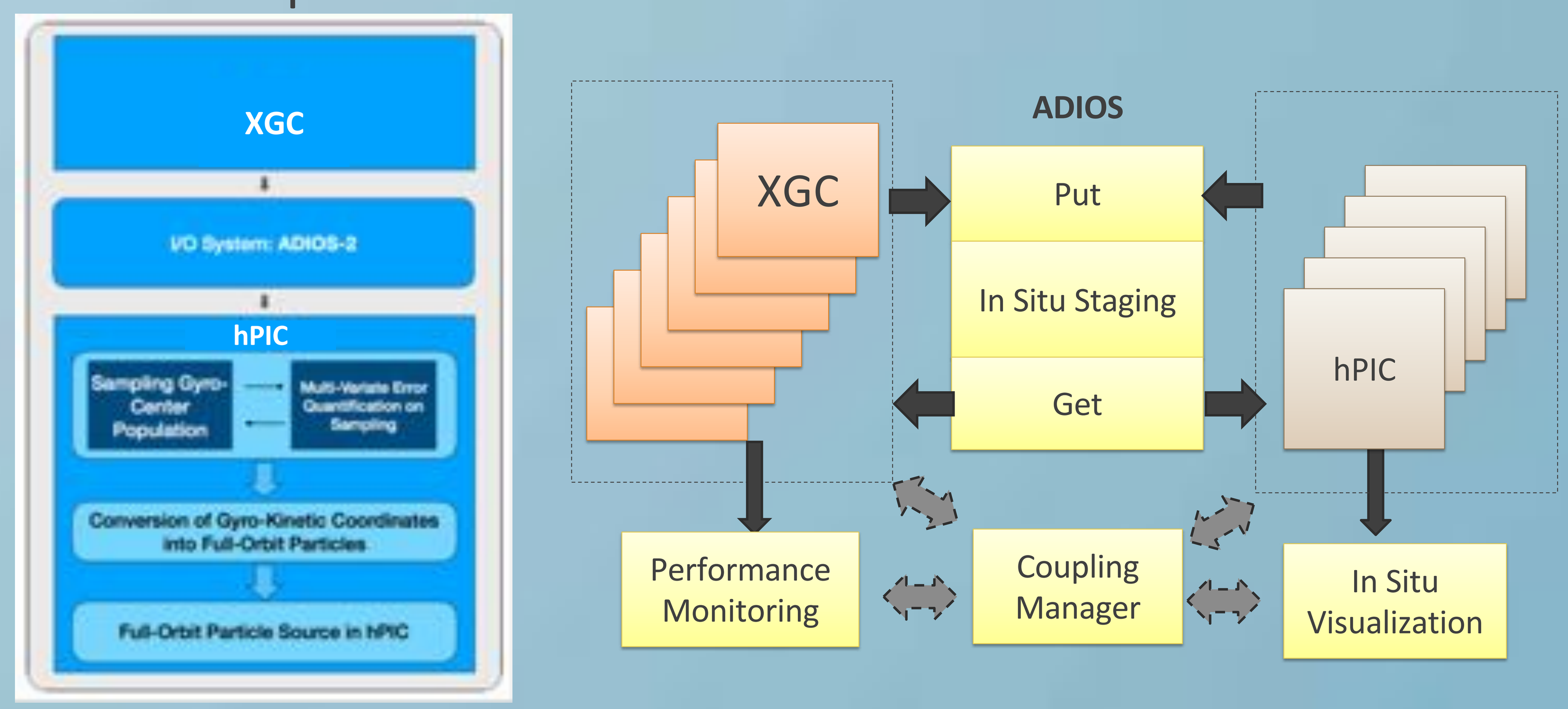
Agile XGC development
- Incorporate a modern CMake build system
- Continuous Integration testing system
- Git workflow incorporated with CI system
- Integrate CDash into github

Coupling Workflows

The Fusion HPBS project is focusing on researching multi-way coupling science to study multi-scale/multi-physics.

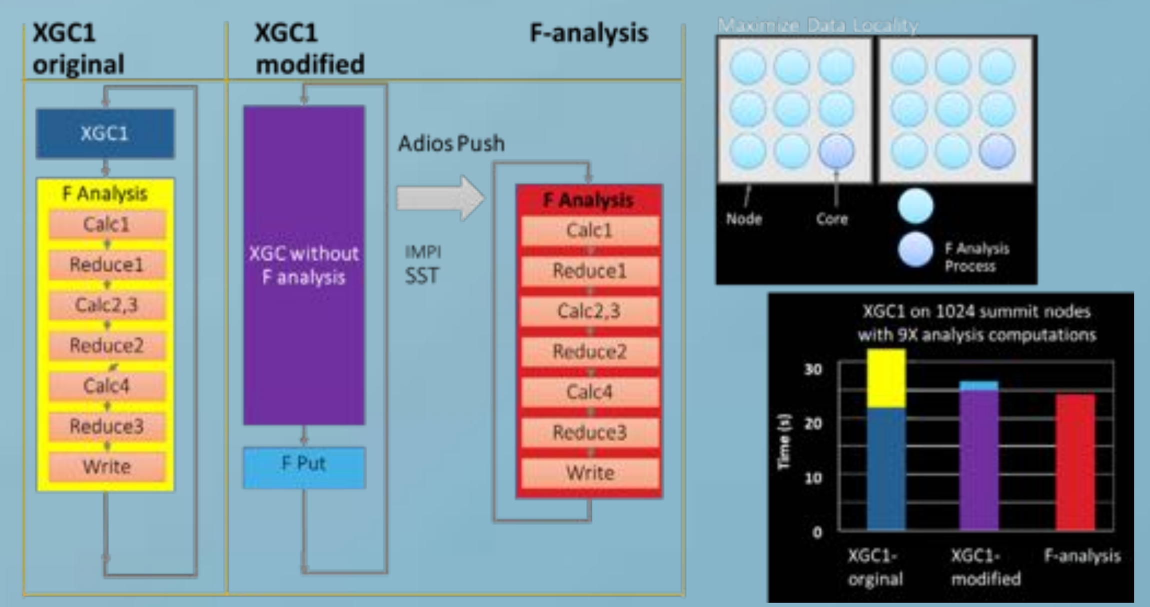
1) XGC and hPIC

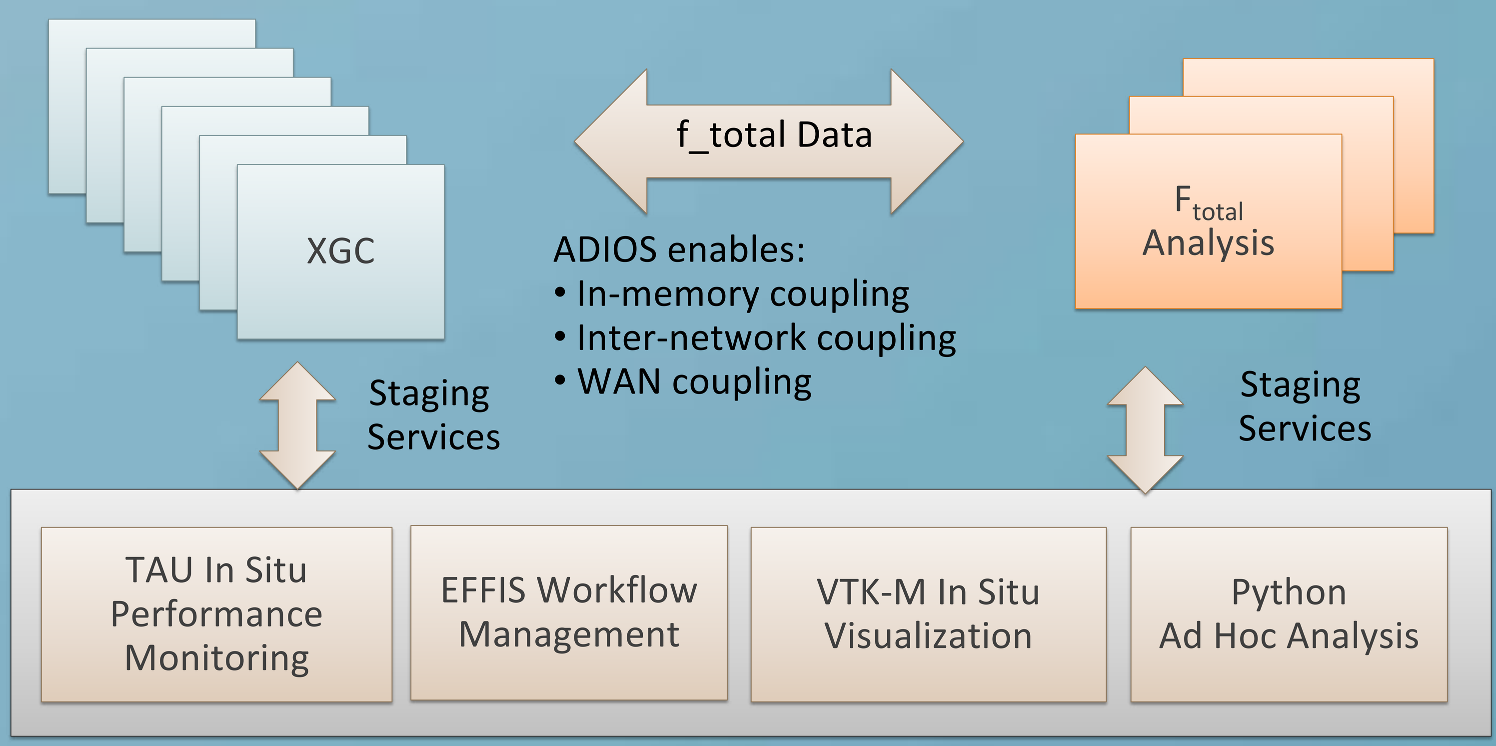
- Plasma-material-interaction hPIC code coupled into XGC
- hPIC code has 6D marker particles, while XGC has 5D marker particles



2) XGC and F-analysis coupling

In XGC and F analysis coupling, we move the F computation to a dedicated analysis code. XGC asynchronously offloads those computations via ADIOS and improves computational performance



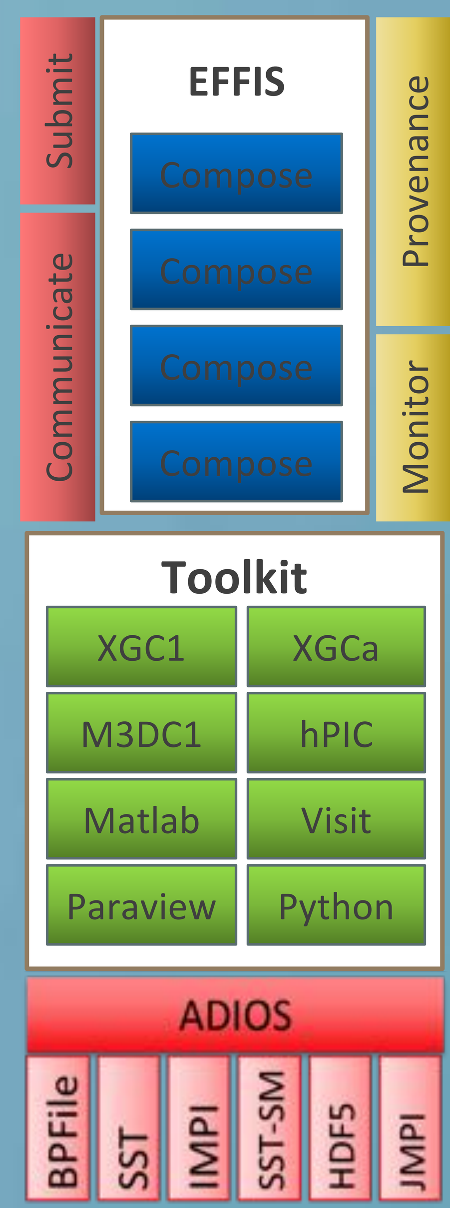


Research Details

a) To improve movement performance and flexibility, HBPS integrated with ADIOS for data management.
b) Developing multi-way coupling science cases to study multi-scale/multi-physics scenarios.
c) Exploiting data locality to improve performance
   - XGC computes 5D *f* and electromagnetic field
   - Hand-off computational reduction of physics from XGC
   - Analysis code consumes in-memory f data

EFFIS

EFFIS is an integrated platform of services to compose, launch, monitor, and control coupled applications.
EFFIS can simplify the complexity of composing, running, and monitoring applications on HPC systems.
We integrate **HBPS** with **EFFIS** to "easily" compose coupled HBPS workflows on HPC Resources (Cori, Theta, and Summit).
EFFIS's using a python-like interface can allow "easy" integration to visualization tools (Visit, Python notebooks)




```
run:
  xgc:
    processes: 1024
    processes-per-node: 32
    path: xgc-build/xgc1-es
    groups:
      diagnosis.1d:
        plot:
          psi-plot:
            x: psi
            y: i_gc_density_1d
```

Execution Management

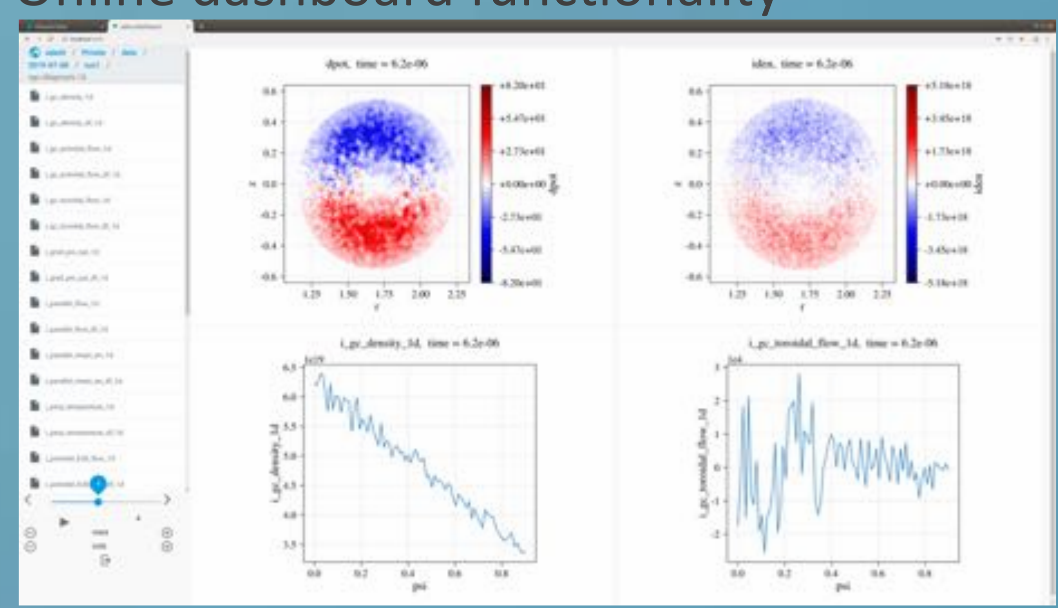
Analysis/Visualization Integration

Example of EFFIS specification file. XGC run is configured to run with analysis application.

```
//@effis-begin reader_io-->"ConcentrationData", step=kstep;
adios2::IO reader_io = ad.DeclareIO("SimulationOutput");
adios2::IO writer_io = ad.DeclareIO("PDFAnalysisOutput");
```

Example of EFFIS instrumentation in XGC code using simple @effis pragmas.

EFFIS is integration with HBPS can provide:
- High Performant I/O for multiple codes
- Process placement (node sharing, co-location of codes on a node, etc.)
- Online dashboard functionality



- Services for concurrent analysis/visualization
- Run archival (e.g. long-term tape storage)
- Source code association with runs

Acknowledgments: Work supported by U.S. DOE Office of Science, ASCR and FES. This research used resources of OLCF, ALCF, and NERSC, which are DOE Office of Science User Facilities.

Performance Enhancements of XGC

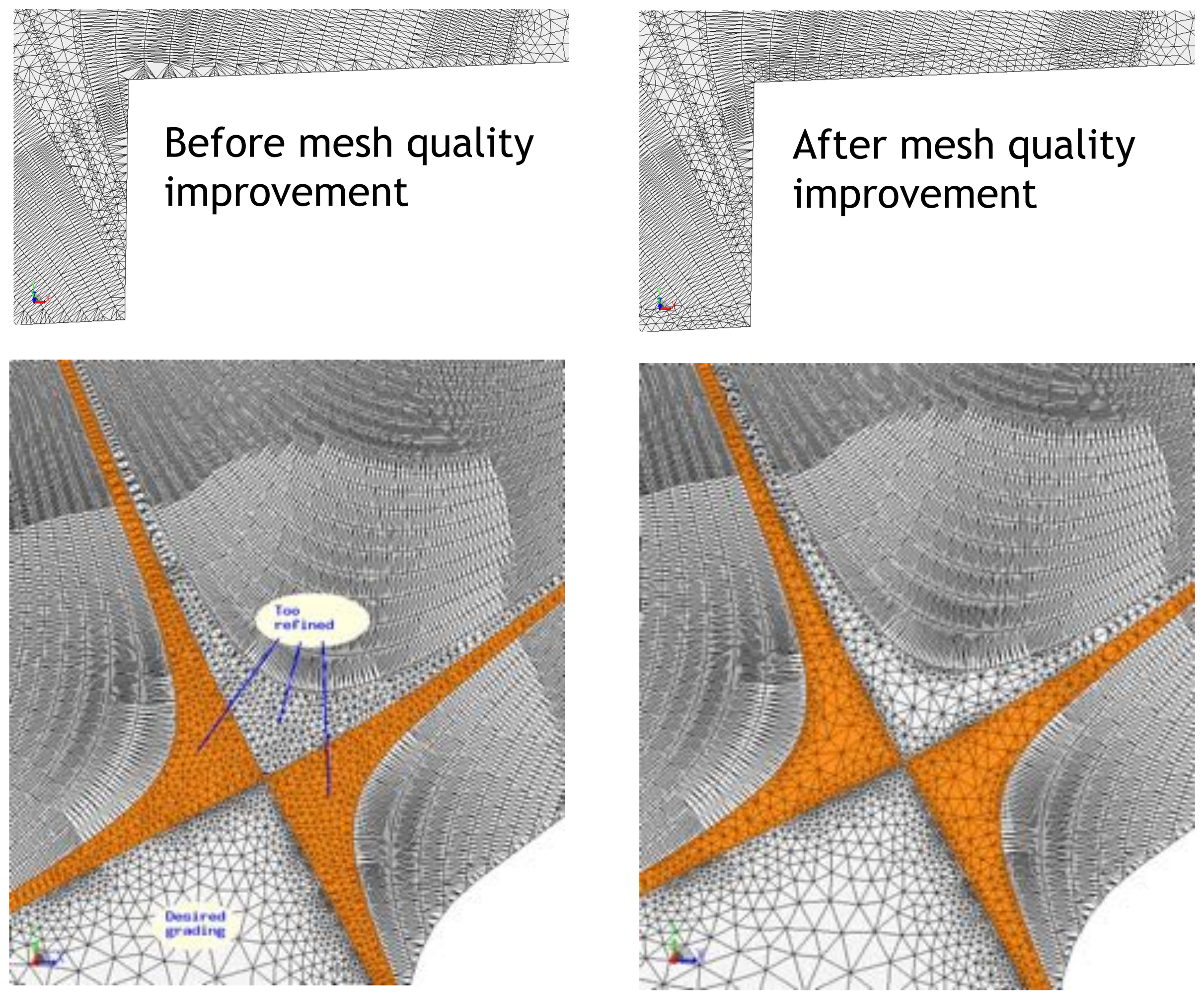
HBPS
High-fidelity Boundary Plasma Simulation

E. D'Azevedo[1], A, Scheinberg[2], M. Shephard[3], P. Worley[4], S. Sreepathi[1], B. MacKie-Mason[5], T. Willians[5], and the SciDAC HBPS XGC Team

1. Oak Ridge National Laboratory , 2. Princeton Plasma Physics Laboratory, 3. Rensselaer Polytechnic Institute, 4. PHWorley Consulting, 5. Argonne National Laboratory
Funding is from DOE ASCR and FES Offices

XGC Meshing

- Improved mesh quality in areas where flux curves interact with reactor wall
- Improved matched mesh gradation at x-point
- Reordering of mesh data for better memory access during XGC simulations

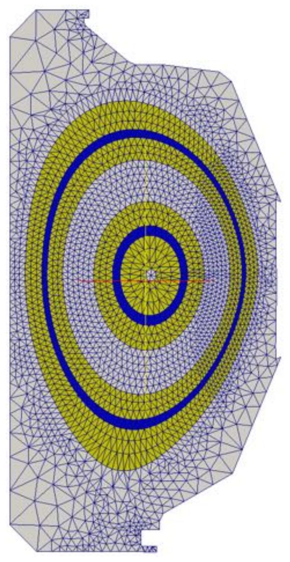


Improved mesh gradation at X-point

XGC based on Parallel Unstructured Mesh PIC (PUMIpic)

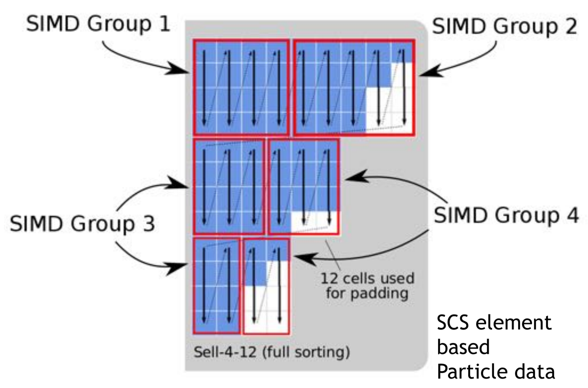
PUMIpic - Components to support PIC operations on distributed unstructured meshes (2D and 3D)

- Mesh centric – no independent particle structure
- Distributed mesh with overlaps (PICparts)
- Particle migration and load balancing between pushes
- Adjacency-based particle containment determination
- Focused on structures for execution on GPUs
  - Omega GPU ready mesh topology being integrated
  - Particles stored by element in new SCS data structure
  - Test shows on-par performance using less memory



Two PICparts

	no sorting	full sorting
ptcls (Ki)	time (s)	time (s)
128	2.298661	3.642041
256	2.895464	3.415048
512	3.79263	3.851178
1024	4.972283	4.090044
2048	7.089673	4.389198
4096	11.578984	4.799475



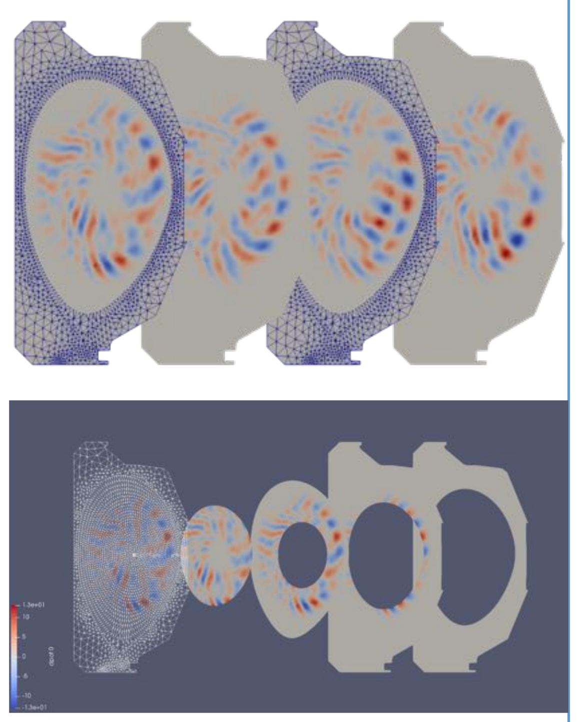
SCS element based Particle data structure

Implementing XGC physics and Numerics with PUMPIpic:

- Since all core data structures are changed code, code being rewritten in C++

Status of implementation:

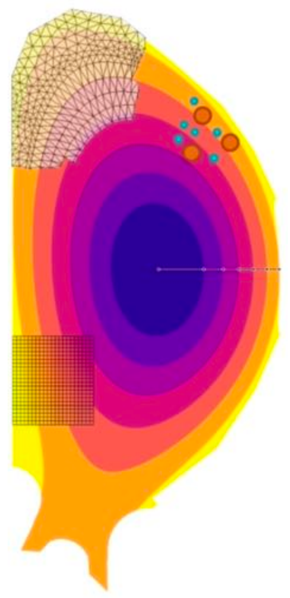
- Based on original PUMI structures - new GPU focused structures will be integrated when complete
- Core mesh/particle interaction operations in place
- Mesh solve in place
- Ion and electron push (including subcycling) implemented
- Initial δf simulations executed
- Performance evaluation and improvement underway
- Initial push results show 25% improvement on many core system
- Other steps slower due to need to modify mesh copies (underway)



Snapshot of electrostatic potential fluctuation (a) at toroidal angle ζ=0,π/2,π,3π/2 from left to right and (b) in local domain of each group at ζ=0

XGC on Summit

- XGC is part of Early Science Programs on Summit, Aurora and Perlmutter
- XGC is an ECP code
- XGC uses an unstructured grid in poloidal plane, each MPI rank gets particles from a section of poloidal plane
- Main computational kernel is electron push
- Utilizes Kokkos

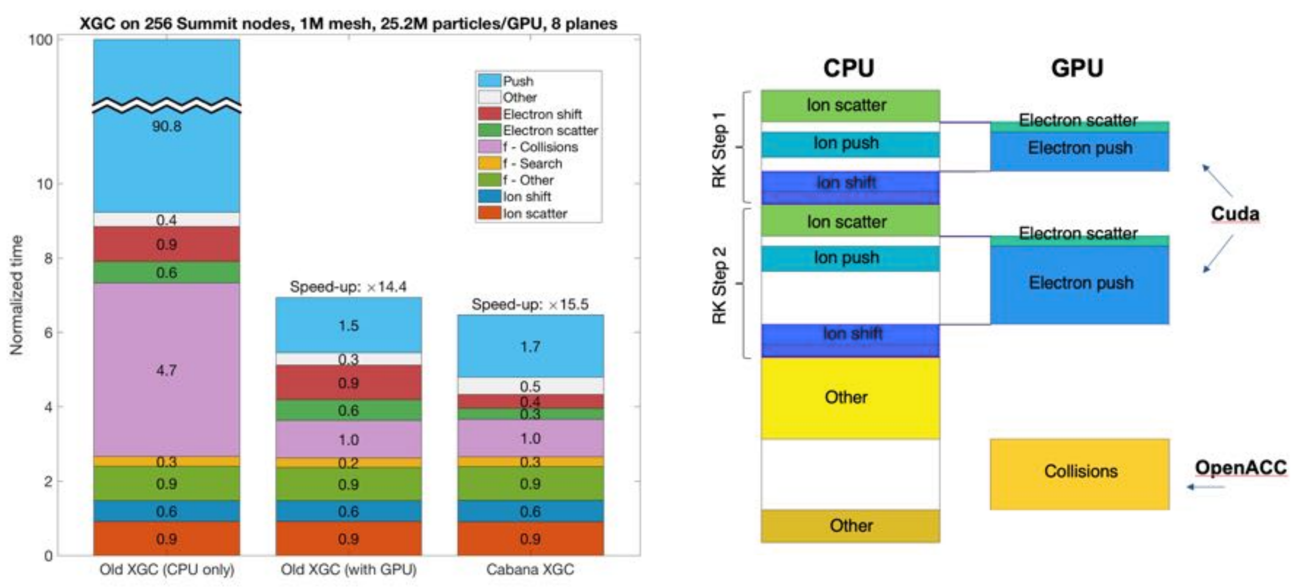


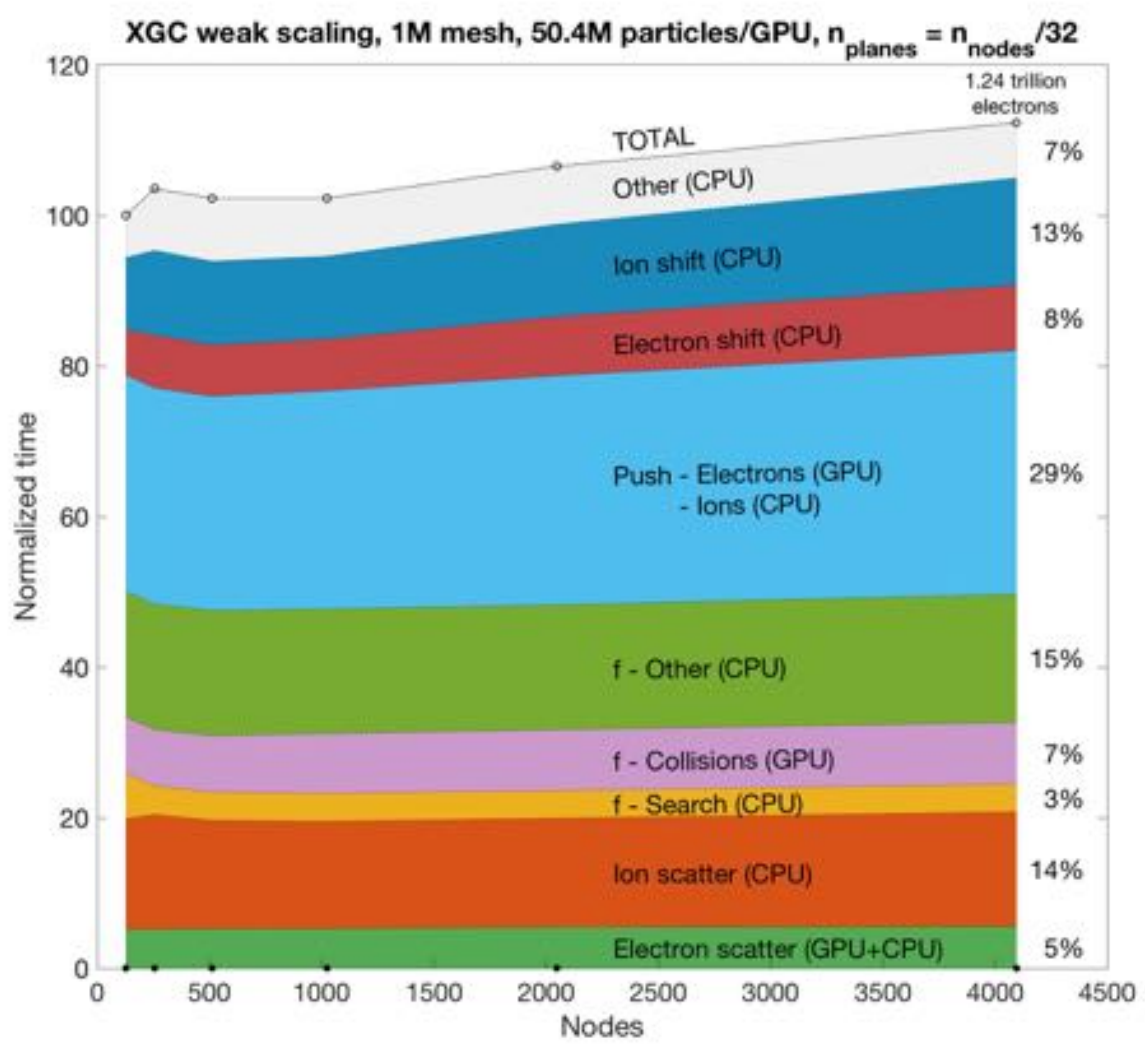
XGC_core/pushe.F90:

```
subroutine pushe
  call sort_particles            ! Sort particles by grid cell
  do iptl=1, n_particles         ! Loop over particles
    do ic=1, n_cycles            ! Subcycle electrons
      do irk=1, n_runge_kutta    ! RK4 loop
        call search                          ! Determine which grid cell particle
inhabits
        call gather_field        ! Interpolate field at particle location
        call calculate_dx        ! Solve physics: dx/dt = f(E,…)
        call advance_particles   ! Update particle position and velocity
      end do
    end do
  end do
end subroutine pushe
```

Good Weak Scaling to Full Summit

- On 256 nodes of Summit, GPU version has **15X speedup over CPU only**
- Good weak scaling up to full Summit using **1.24 trillion electrons on GPU and 1.24 trillion ions on CPU**





Details on XGC-Kokkos

- XGC in Fortran, Kokkos in C++
- Fortran interface (Cabana) enables easy porting of new kernels
- Single code for CPU and GPU
- Electron push kernel in CUDA Fortran (C++ version under development)

```
void main()
{
  Cabana::initialize();

  // Create instance of array of structure of arrays
  ParticleList particles( num_particle );

  // Create "range policy"
  Cabana::RangePolicy<ParticleList::array_size,ExecutionSpace> range_policy( 0, particles.numSoA() );

  // Main time loop
  for (int i=1; i<=n_steps; i++){
    ... // Deposition, field solver, etc.
    Cabana::parallel_for( range_policy_vec, push_electrons, Cabana::IndexParallelTag() );
  }

  Cabana::finalize();
}

// Electron push subroutine is a lambda function
auto push_electrons = KOKKOS_LAMBDA( const int idx )
{
  ...
};
```

```
num_vecs=num_particle/vector_length
... ! Platform-specific directives etc.
do i_vec=1,num_vecs
  call push_electrons(all_particles(i_vec),i_vec))
end do
```

```
! Macro generating Fortran INTERFACE
PARTICLE_OP_F(pushe)

subroutine pushe_f(particle_vec, i_vec) BIND(C,name='pushe_f')
  USE, INTRINSIC :: ISO_C_BINDING
  type(ptl_type) :: particle_vec
  integer(C_INT), value :: i_vec
  do i=1, vector_length
    ... ! Vectorizable loop that advances particle positions
  end do
end subroutine
```

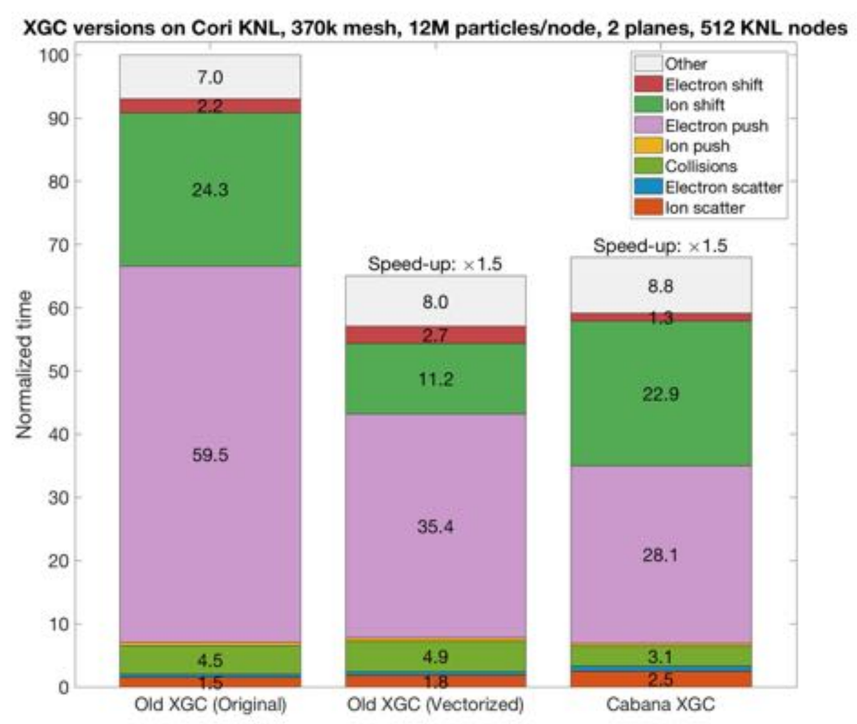
Must cast Cabana array into predefined Fortran type for use in Fortran kernels using ISO_C_BINDING

```
module ptl_module
  use, intrinsic :: ISO_C_BINDING
  type, BIND(C) :: ptl_type
     real (C_DOUBLE) :: ph(vector_length,6)
     real (C_DOUBLE) :: ct(vector_length,3)
     integer (C_INT) :: gid(vector_length)
  end type ptl_type
end module
```

```
// Create Cabana structure type
using ParticleDataTypes =
        Cabana::MemberDataTypes< double[6], double[3], int >;

using ArrayLayout =
        Cabana::InnerArrayLayout<vector_length,Cabana::LayoutLeft>;
using ParticleList =
        Cabana::AoSoA<ParticleDataTypes,MemorySpace,ArrayLayout>;

// Create instance of array of structure of arrays
ParticleList particles( num_particle );


// Create analogous C structure of arrays
struct local_particle_struct {
    double ph[6][vector_length];
    double ct[3][vector_length];
    int gid[vector_length];
};

// Define array of pointers to particle structures
auto* p_loc = (local_particle_struct*)(particles.ptr());
```

Performance on KNL

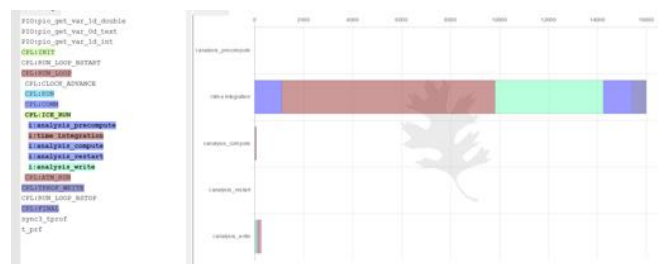
- Kokkos version of XGC has been ported to Cori KNL
- Roofline analysis of vectorized version of XGC shows in-lining and re-factoring useful in optimizing use of wide-vector registers. However, vector dependences and data type conversions limiting peak performance



Performance Analytics for Computational Experiments for XGC

- Central hub of performance data, already used in Climate application
- Interactively deep-dive and track performance benchmark
- Facilitate performance analysis:
  - Load balancing
  - Identification of bottlenecks
  - Inform targeted optimization efforts

https://pace.ornl.gov



Tree and Flame Graphs