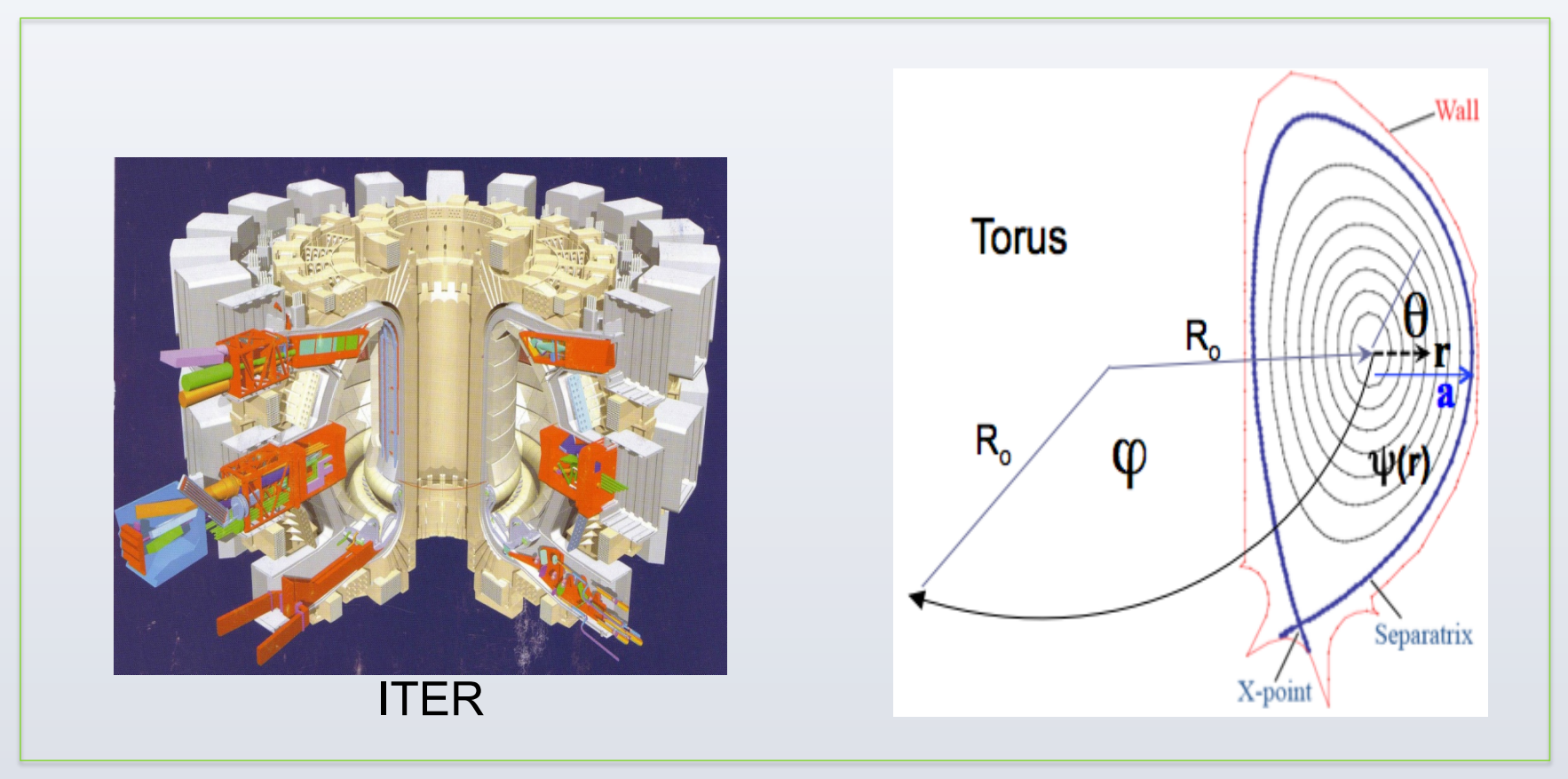
Enabling Science and Technology in the Center for Edge Physics Simulation

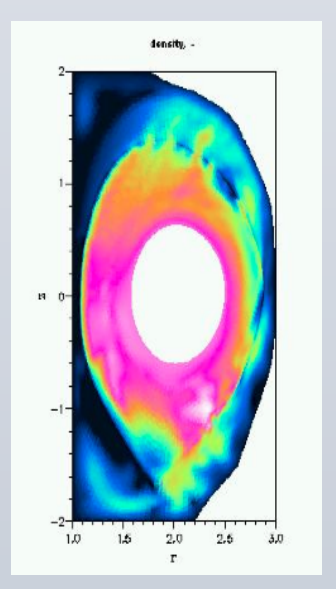
M.S. Shephard[1,a], M.F. Adams[2,a], E. D'Azevedo[3], J. S. Hesthaven[4], S. Klasky[3,b], R.D. Moser[5,c], M. Parashar[6], N. Podhorszki[3], P. Worley[3,d], C.S. Chang[7], S. Ku[7] and the EPSI Team

[1]RPI, [2]Columbia U. in transit to LBNL, [3]ORNL, [4]Brown U., [5]U. Texas, [6]Rutgers U., [7]Princeton Plasma Physics Laboratory, ([a]FASTMath, [b]SDAV, [c]QUEST, [d]SUPER)

Edge Physics Simulation



- Edge plasma conditions determine (i) core plasma quality, thus the fusion efficiency, and (ii) wall deterioration, thus the reactor lifetime
- Unlike the core plasma, the edge plasma contains magnetic separatrix and material wall → Non-equilibrium thermodynamics.



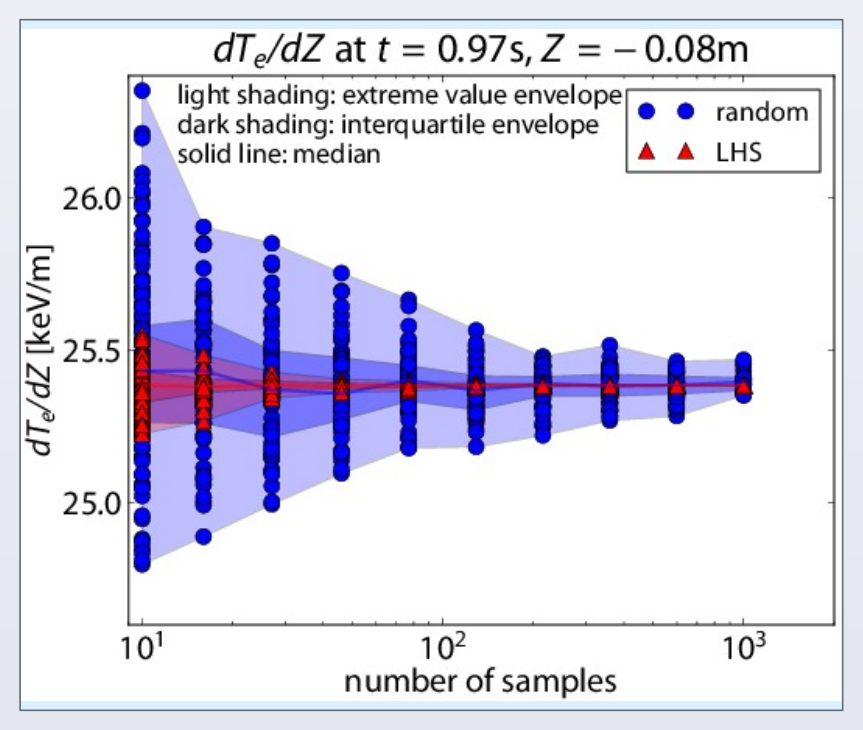
EPSI Center's massively parallel XGC codes
- Combination of particle- and mesh-based methods
- Have demonstrated excellent scaling on leadership class HPCs, in collaboration with SUPER
- Code development supported by key applied math and computer science advancements
- Integration with uncertainty quantification

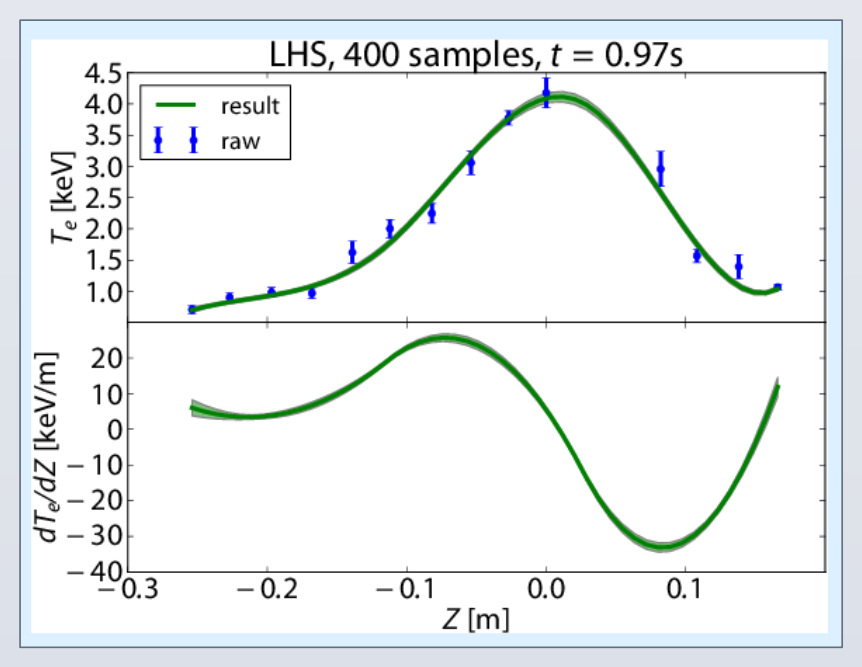
When edge pedestal becomes too steep, edge localized instability deteriorates wall.

UQ Analysis Plan

We are introducing key UQ methodology into the simulation workflow, using DAKOTA and QUESO.
- Improved UQ in derived XGC1 inputs and validation observables: Sampling methods for profile smoothing, EFIT and TRANSP outputs
- Forward sensitivity analysis of temperature and density profiles to key XGC1 inputs (heating power, neutral recycling rate, magnetic field geometry) and physics choices (various turbulence modes, impurity)
- Bayesian calibration of reduced-physics models (anomalous transport model in XGC0)

Profile Smoothing/Fitting





- Apply UQ not only to XGC1, but also to experimental validation data
- Automated profile smoother, using bivariate space-time splines with sawtooth binning (AIC, BIC for spline properties)
- Use DAKOTA to benchmark MC and Latin Hypercube Sampling (LHS) profile fitting
- LHS showed improved convergence rate, variance reduction
- Reducing sample size important when extended to more expensive "samples"- outputs from EFIT or TRANSP
- Currently under extension to other codes, sampling methods (ILHS)

Challenges: Poor uncertainty models for inputs to diagnostic routines, incorporating data from other diagnostics into fitting routines, core and edge constraints

XGC1 Sensitivity Analysis

Currently we are evaluating sensitivity of key 1D physics profiles (T_i,n_i) and their gradients to model parameters (heating and cooling) and numerics (timestep size, particle number, spatial grid size).

- Dedicated UQ branch of XGC1 software repository with access to main branch XGC1 routines, scripting support for interface with UQ tools
- Currently exploring simplified ITG physics as a reduced model allowing sufficient UQ sample size
- Incremental process of adaptively enriching the model when indicated by experimental data

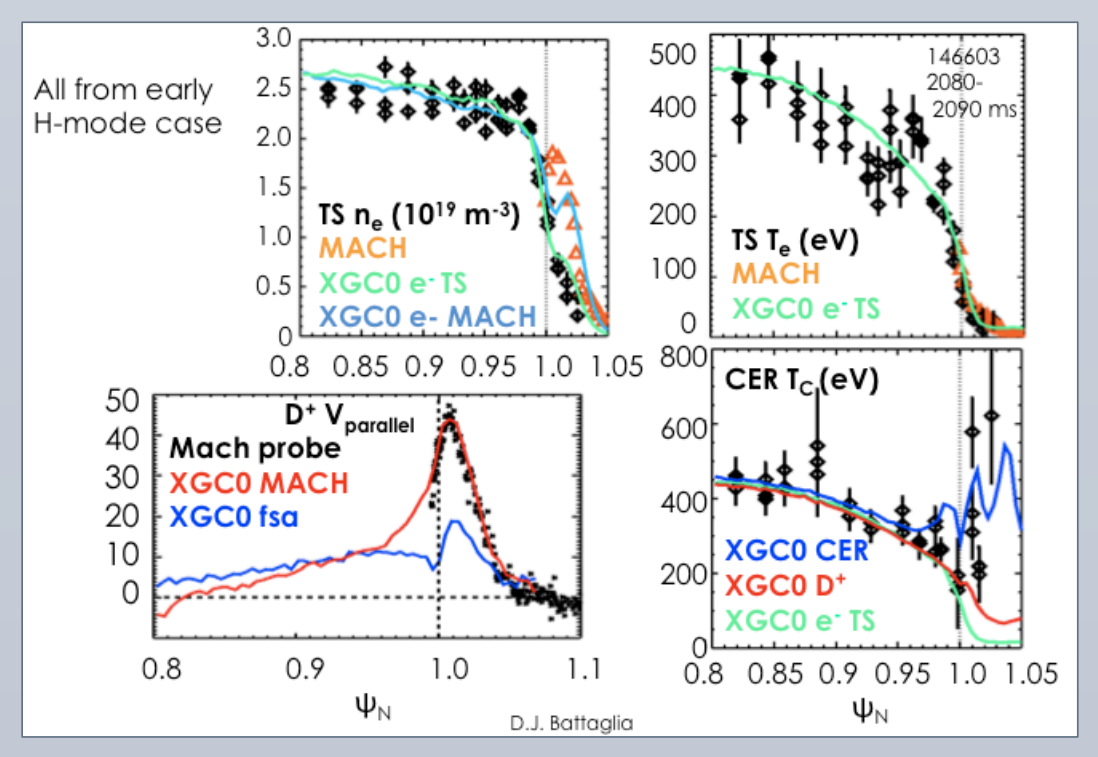
Challenges: Balancing computational demands of full-physics simulation versus sampling demands of UQ, developing UQ analysis to extrapolate from simplified to more complex physics models, treatment of sampling bias due to numerical artifacts

Calibration of Reduced Model in XGC0

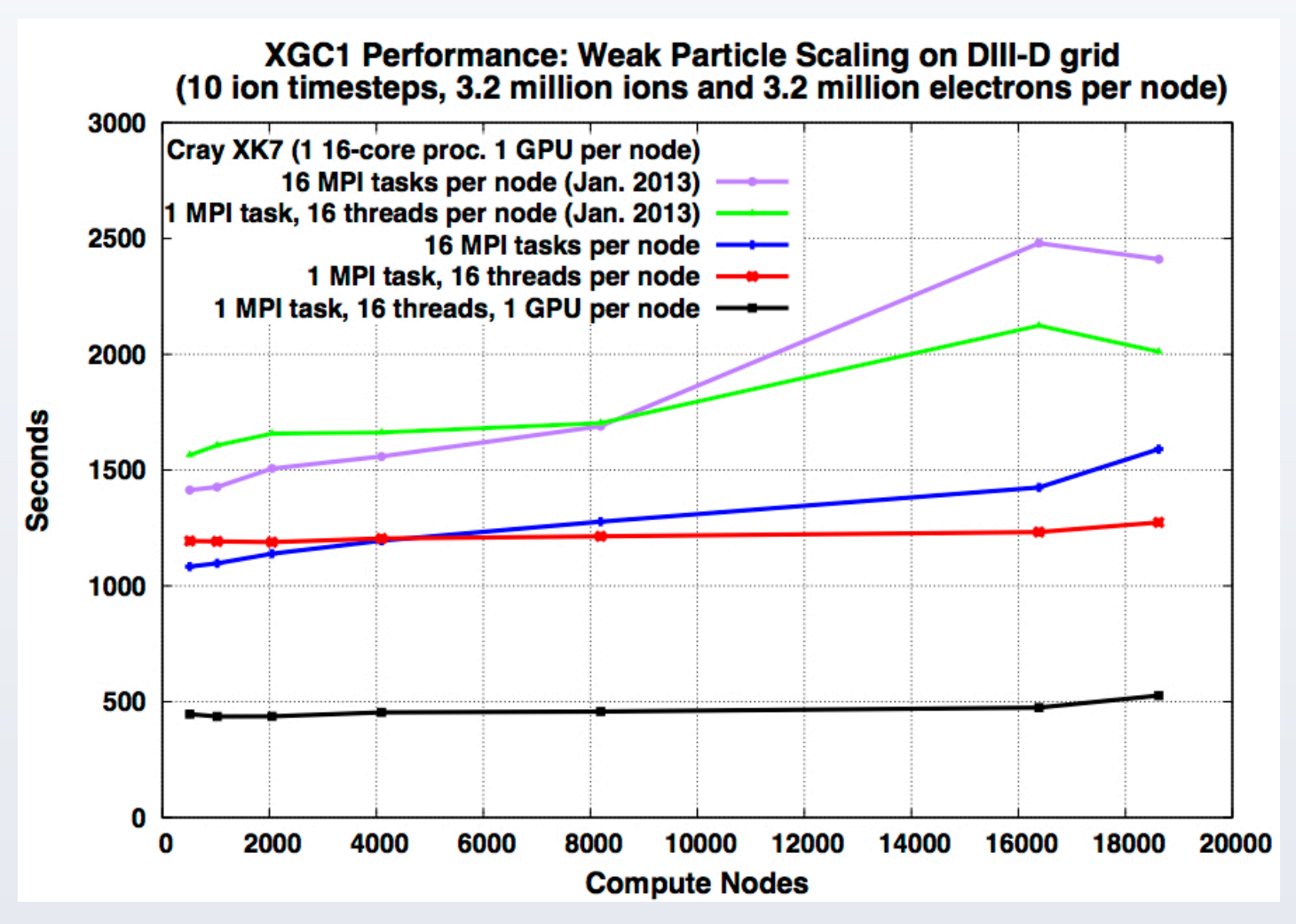
- **Goal**: Bayesian calibration of anomalous transport model using H-mode DIII-D data in reduced physics (XGC0) using QUESO
- Similar process potentially applicable to calibration of XGC1 inputs

Challenges: General XGC1 extension may lead to a large, expensive inverse problem

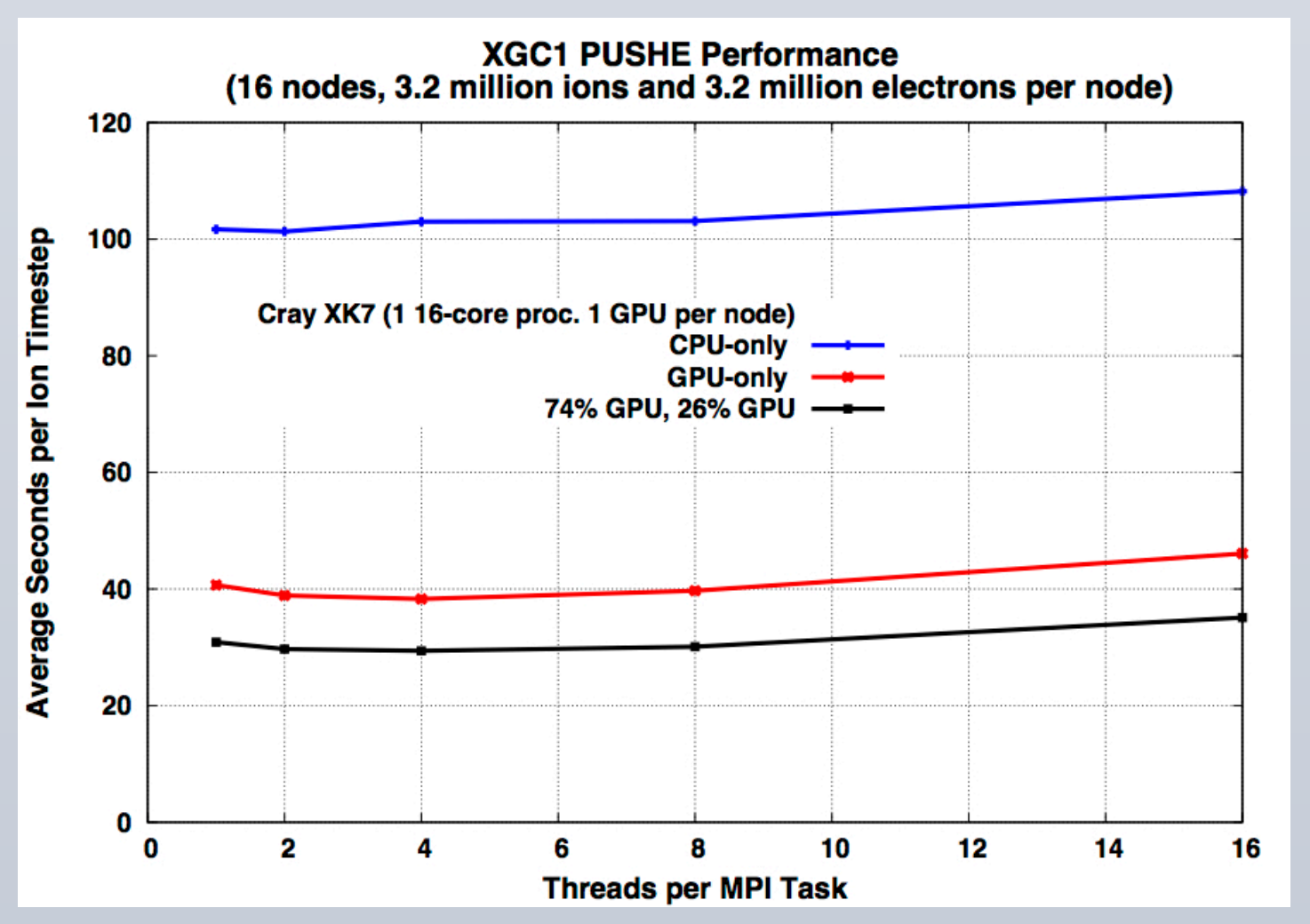
Manual calibration of 5 parameter anomalous transport model yielding promising results (D. Battaglia)



4X Performance Improvement at Scale



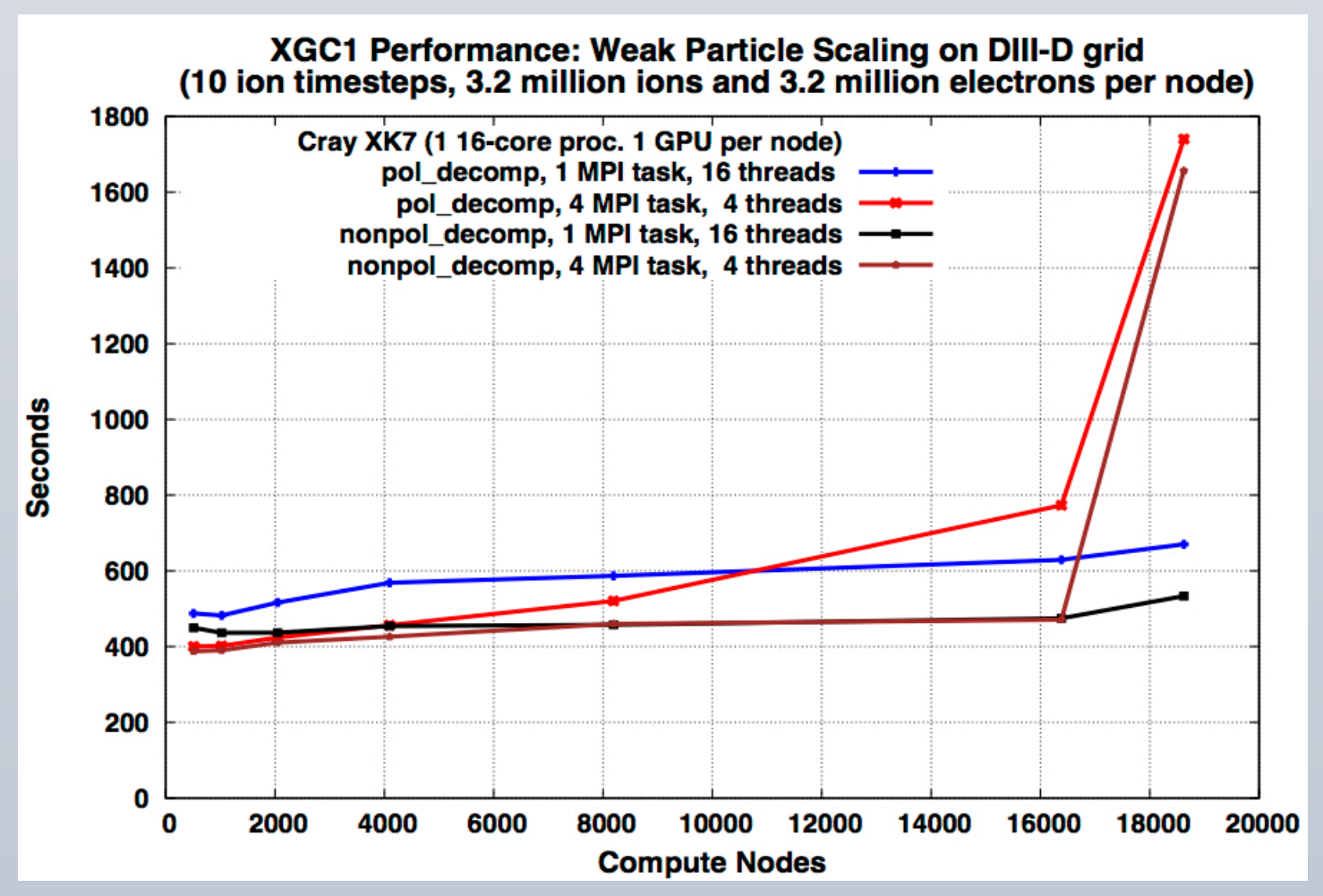
Optimization of distributed memory and shared memory parallel algorithms and porting of computational kernel to GPU accelerator improved computer performance over the January 2013 CPU-only version by a factor of between 3.5 and 4.5 for problem sizes of scientific interest (>= 8192 compute nodes in weak scaling study) on Cray XK7 Titan at the Oak Ridge Leadership Computing Facility.
- The computational kernel for simulating trajectories for electrons (PUSHE), which accounts for > 85% of the overall run time for the optimized CPU-only version, is the initial target for optimizing execution using the GPU
  - GPU kernels are generated using the PGI CUDA Fortran compiler, while OpenMP is used to exploit parallelism on the multi-core CPU.
  - The particle workload is partitioned between the GPU and the multi-core CPU on each compute node. Using just the GPU for PUSHE is between 2.5 and 2.7 times faster than using just the multi-core CPU. An assignment of 74% of particles to the GPU and 26% to the CPU cores optimizes performance, and is between 3.1 and 3.5 times faster than using just the multi-core CPU.



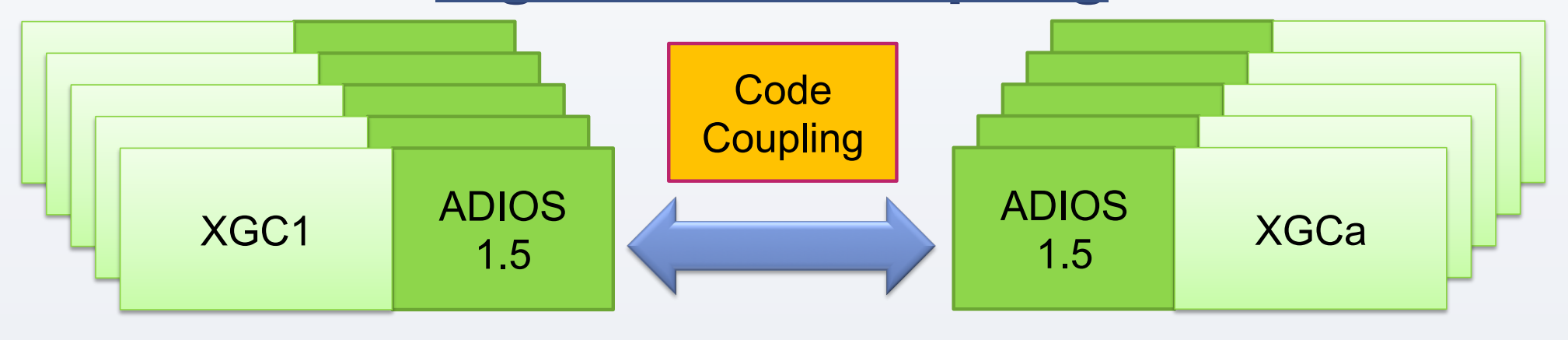
- The optimal number of OpenMP threads to use per MPI task must take into account the MPI overhead and full-code OpenMP efficiency, but up to 8 threads can be used with little degradation of PUSHE performance. 16 threads currently degrades PUSHE performance by 19%.

Performance: Next Steps

- Electrons have higher velocities than ions and can cross multiple subdomains in each ion time step. To enable PUSHE to proceed without MPI communication, the global electric field is replicated on each GPU. For high resolution electromagnetic simulation of ITER, this will be difficult. The memory requirement can be reduced by a factor of 6 if only the potential is replicated and the field components are computed on the GPU as needed.
- To avoid collisions in update operations, certain arrays are replicated. On the GPU this leads to large memory requirements when thousands of threads are used. Alternative implementations are being developed that exploit the recently available efficient atomic update operations on 64-bit floating-point values, lowering memory requirements and allowing more threads to be launched, hopefully improving performance.
- New science capabilities will require a 2D domain decomposition to partition grid and particles ("poloidal decomposition") instead of the current 1D domain decomposition and random partition of particles in other dimensions. New capabilities will change the performance characteristics significantly, but experiments using the current version indicate a computational load imbalance on the GPU not related to imbalance in number of particles. Non-power-of-two MPI collectives also appear to demonstrate poor performance at scale for multiple MPI tasks per node.



Tight Code Coupling



ADIOS (2013 R&D 100 winner) has been developed to provide low-latency code coupling in EPSI simulation to support tightly coupled execution scenarios. The main focus of the development is to provide:
- Low-latency, tight coupling execution environments through memory-to-memory data exchanges between different codes/executions
- Service Oriented Architecture (SOA) for on-demand coupling executions with support of dynamic workflow invocation
- Coupling executions in heterogeneous computing environments

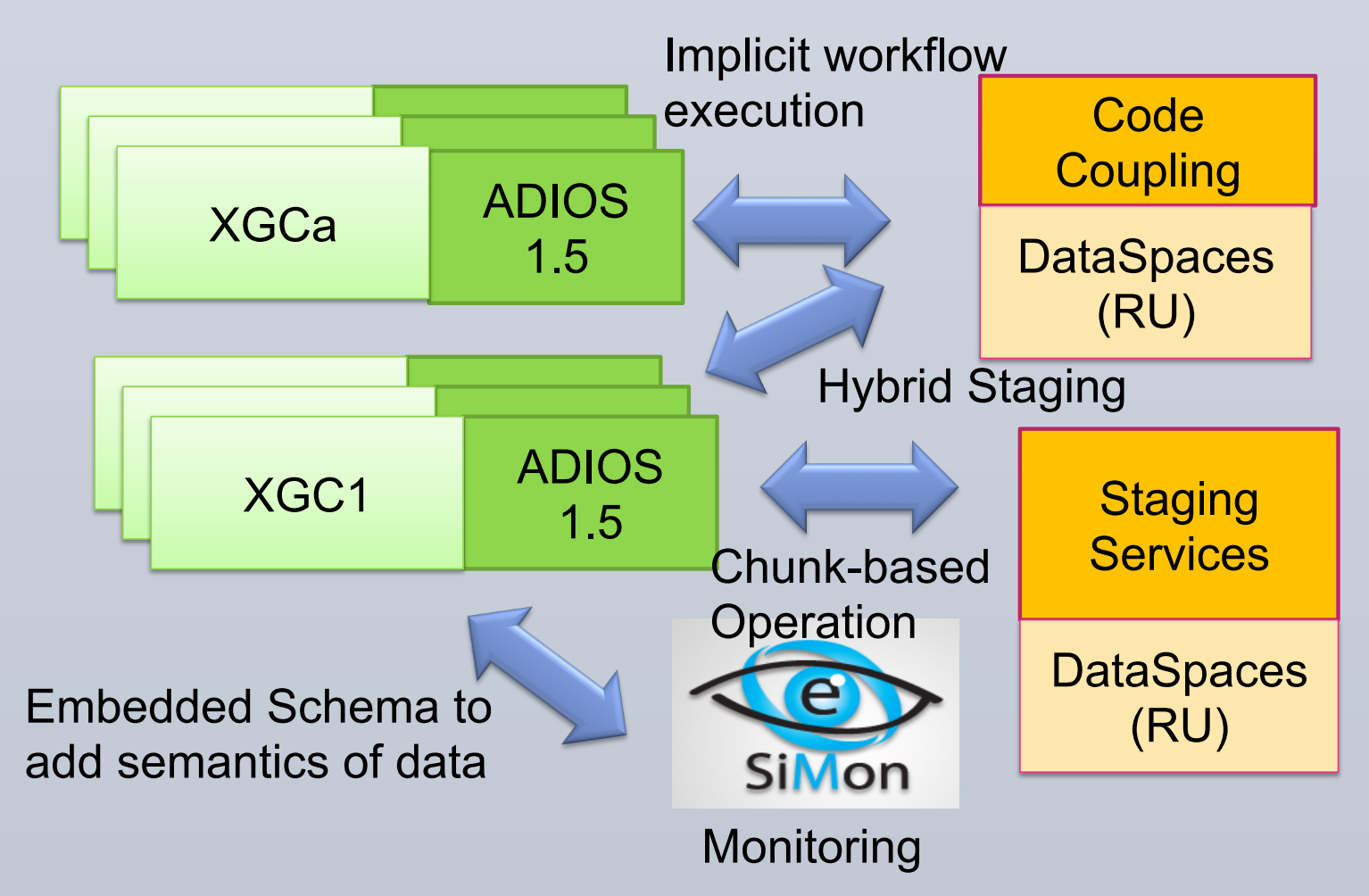
Evolution of coupling

Past approach: Kepler + ADIOS
- File-based code coupling resulted in high latency, low-throughput, and under-utilization of resources.
- Static workflow demanded lots of human efforts in designing and updating workflows.
- Customized applications with hard-coded execution plan.

Paradigm shift: ADIOS + embedded workflows
- Support efficient memory-to-memory/in-memory multi-code coupling (e.g., using DataSpaces).
- Semantic-rich, machine-readable information embedded in data requires less human involvement and provides more efficiency and flexibility in coupling execution.

Data-centric integrated execution environment

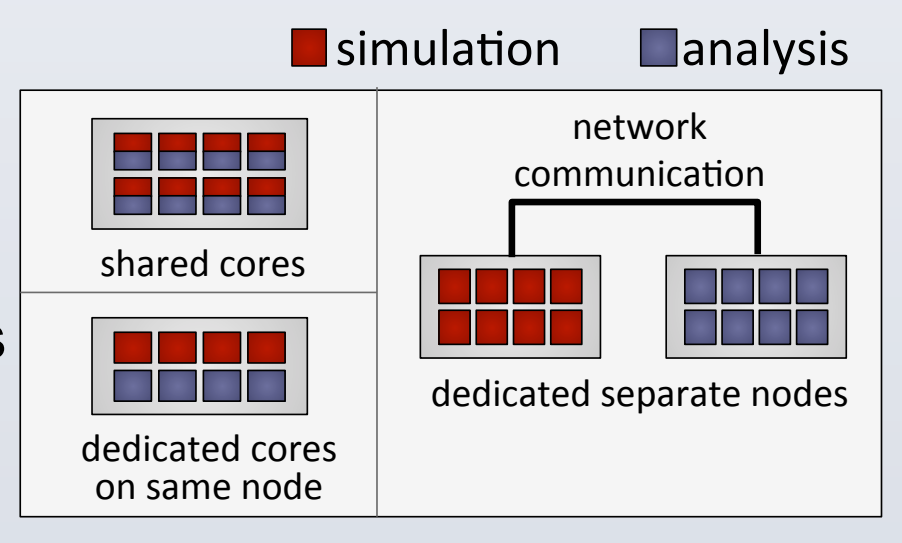
Our focus is to support EPSI by providing integrated data-centric execution environments for tight code coupling, staged data process, and monitoring system with a support of dynamic workflow system



Data Staging

Recent ADIOS release (version 1.5) with DataSpaces has been incorporated in EPSI simulations to support efficient I/O operations:
- Streaming data
- Non-blocking operations
- Selection and chunked reads to enable schedule optimization
- Staging with a unified API set for file and in-memory coupling
  - Maintain backward compatibility
  - Read data from files or memory with a unified API

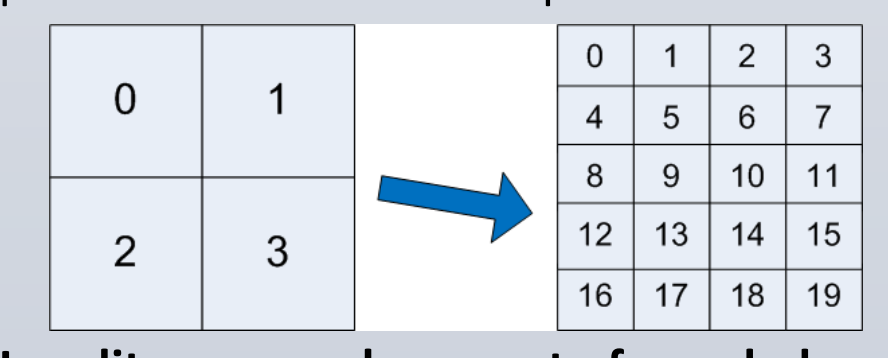
Hybrid staging



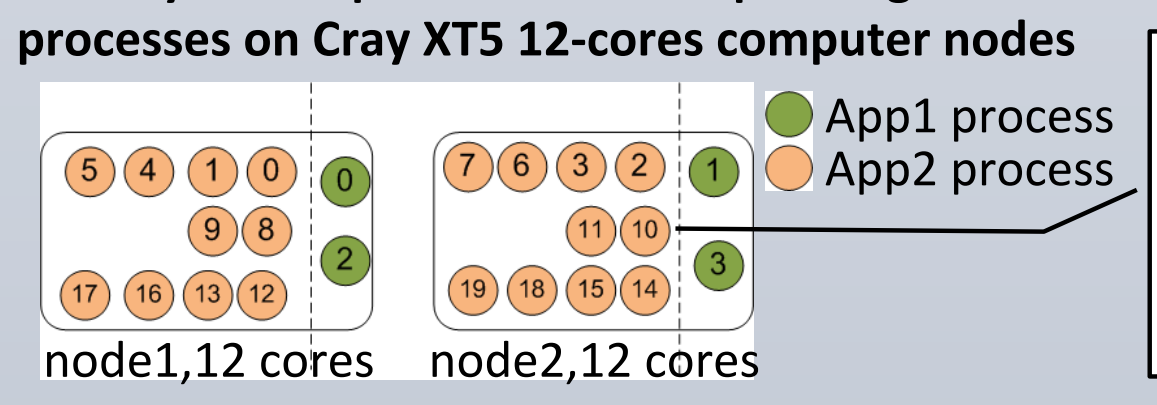
- Hybrid approach to deal with exploding data volume
- Open questions on using GPUs and SSDs
- Asynchronous decoupled analysis for faster time to solution
- Enables online in-situ/in-transit data processing, and asynchronous memory-to-memory data sharing for coupled simulation workflow

App1 coupled region distributed over 4 processes

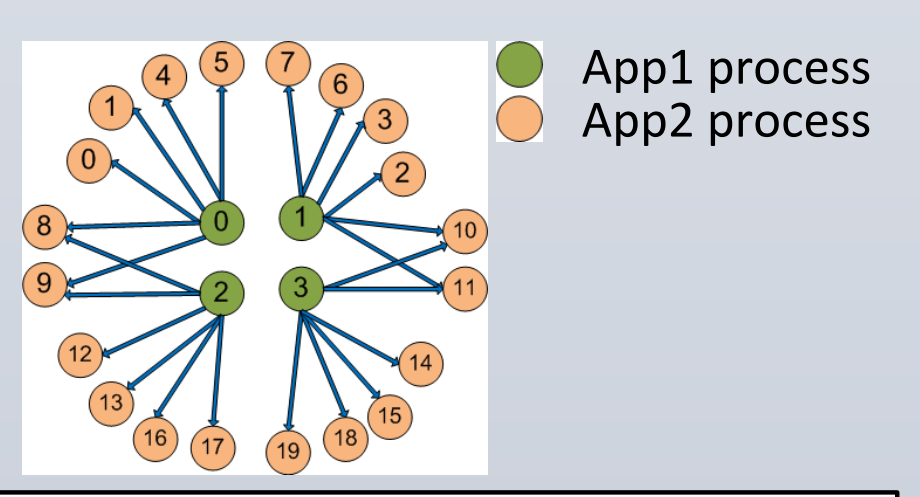
App2 coupled region distributed over 20 processes



Locality-aware placement of coupled region processes on Cray XT5 12-cores computer nodes



Graph of data transfers between coupled codes

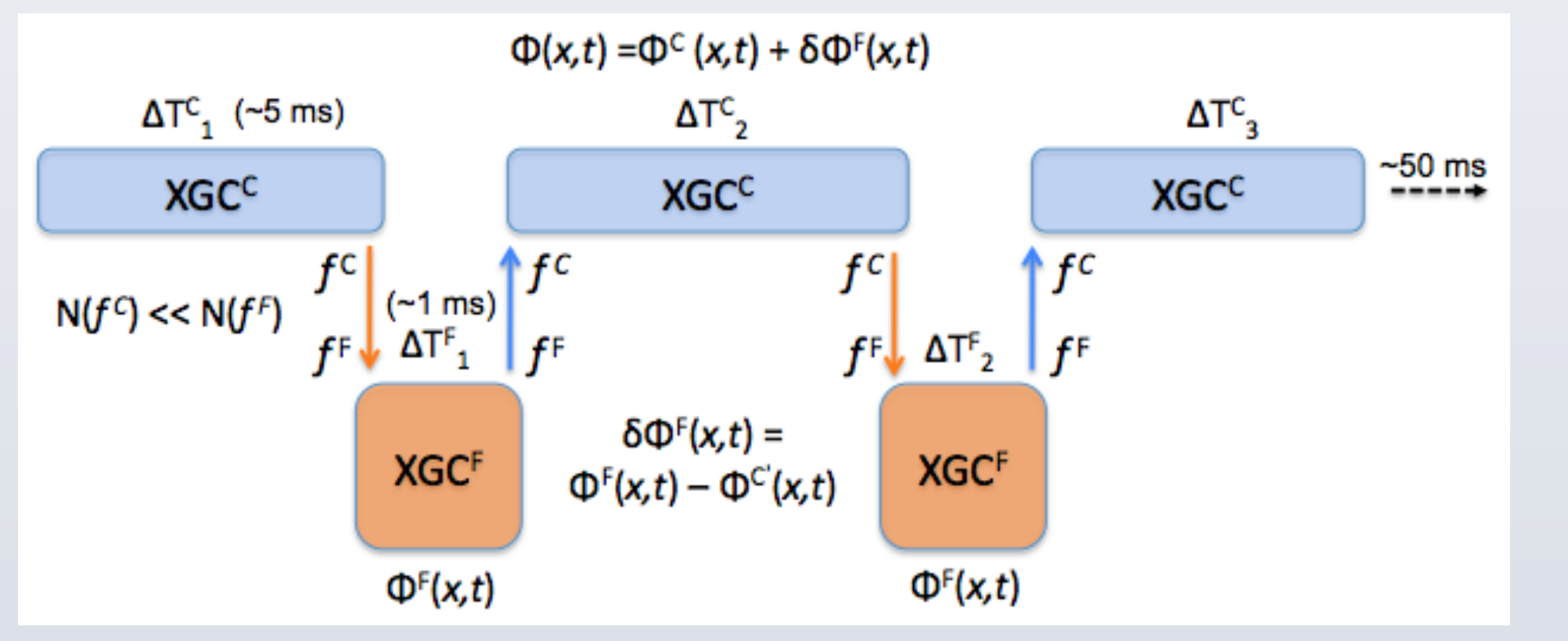


1) Inter-node coupling data transfers (over networks) is minimized
2) Intra-node coupling data transfers can be performed using more efficient shared memory

- Data-centric placement (figure above) enables in-situ execution for EPSI coupled simulation, thus increase the amount of in-situ intra-node data sharing and reduce cost (e.g. latency, energy) of network data movement

Multi-scale Time Advancement

- Prolong the high fidelity simulation to experimental time scale (~50 ms)
- Expensive turbulence simulation may not be needed at all time steps
- Reset error accumulation in the way
- Divide XGC1: XGC^F(axisymmetric+turbulence) and XGC^C(axisymmetric)
- Use Φ^F(turbulence) in XGC^C, with updates as needed
- Requires collaboration with Math, DM, UQ, and Optimization scientistists



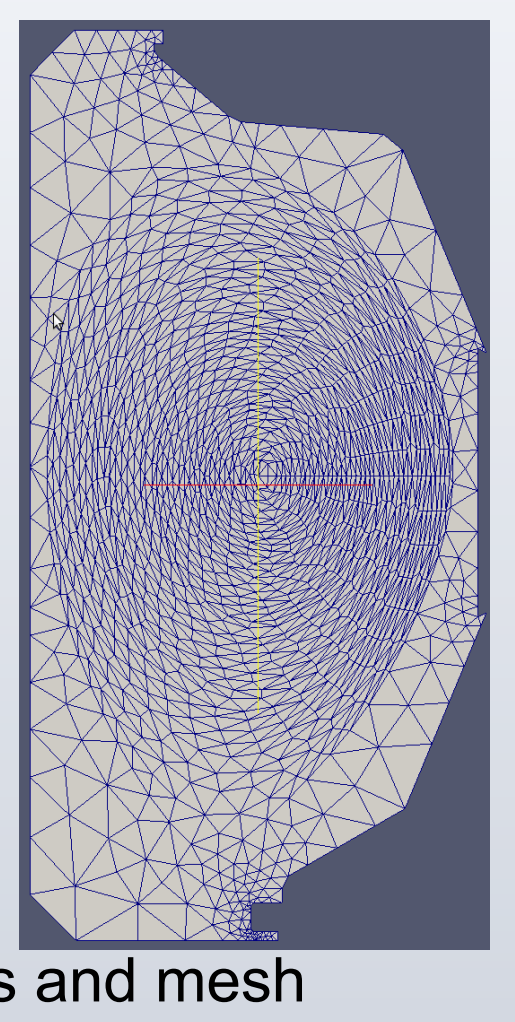
Accomplishments
- Developed a detailed strategy for coarse/fine grain coupling to encode turbulent information in the coarse grained simulation.
- Developed a coarse grained XGCa from XGC1, and demonstrated data coupling with XGC1.
- Identified a strongly turbulent benchmark case to help guide the development of strategies for adaptive multi-scale advancement.

Challenges and next steps
- Identify and understand appropriate measures of quality of the simulation and validate against fully resolved simulations to quantify the impact of the multi-scale approach on the physical fidelity.
- Study physically correct sampling of particles when coupling the two codes to minimize the transitional effects, e.g., phase space density reconstruction and conditional sampling techniques.
- Strategies for stiff profile evolution and solution bifurcation.
- Develop algorithm to determine the coupling time steps.
- Strageties for V&V and UQ.

Solvers

- Scalable Poisson-type Solvers – next steps
  - Continue to develop non-linear solver for Boltzmann electrons
  - Scale linear solver to strong scaling limit for exascale machines
  - Ampere's Law solver with electromagnetic perturbations
  - Higher order interpolation & high-order discretizations
  - Solve full gyrokinetic field equations with flux-surface averaging:
    - Formulate with auxiliary variable and … $-\Delta\phi + (\phi - \langle\phi\rangle) = \rho$
    - Use **PETSc's FieldSplit** solvers
    - Add auxiliary variable for flux surface average: $\begin{bmatrix} -\Delta + I & -B \\ -C & I \end{bmatrix}\begin{bmatrix} \phi \\ \lambda \end{bmatrix} = \begin{bmatrix} \rho \\ 0 \end{bmatrix}$
- FMM-like solvers for screened potential problems
  - FMM-accelerated solver for variable coefficient and non-linear Poisson problems
  - 3D FMM based solvers for electrostatics and electromagnetics
  - Performance optimization on exascale

Unstructured Meshing



Methods and tools for XGC unstructured meshes – Efforts include:
- Generation of better meshes meeting constraints
  - Control of element shapes and gradation
  - Maintain aligned mesh layers between curves of constant flux on interior (and exterior if possible)
  - Introduce increased flexibility around x-point and at geometric features at outer walls
- Parallel mesh and particle
  - Currently have a copy of mesh on each core – potential scaling and memory issue
  - Evaluating using FASTMath PUMI parallel mesh to control parallel mesh and particle methods
- Controlling errors on transfer of data between particles and mesh
  - Error sources include:
    - Linear mesh edges approximating curved flux surfaces
    - Mesh spacing between flux surfaces not assuming piece-wise linear while actual variation is non-linear
    - Piece-wise linear approx. over elements covering many particles
  - Errors inherent to use of different "basis" in two methods
  - Error reduction options – finer and/or higher order (curved) meshes