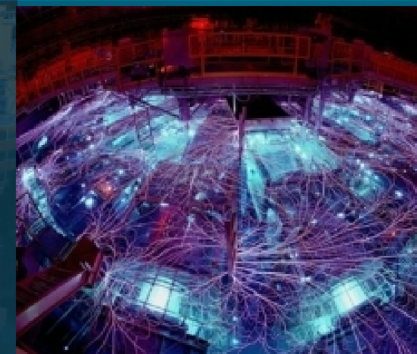
SAND2021-7845C

Sandia National Laboratories

Recent Advances in Maxwell Solvers for PIC Simulations Across Architectures



TRILINOS

Jonathan Hu, Christian Glusa

U.S. Department of Energy NNSA

Platform for Advanced Scientific Computing (PASC) Conference, July 5-9, 2021

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Outline

- Motivating application
- Governing equations
- Discretization
- Algebraic multigrid solver
- Numerical results
- Future work

Motivation

Sandia mission apps are being readied for current and emerging pre-exascale architectures, as well as future exascale architectures

- DOE ASC ATDM program
- DOE Exascale Computing Project (ECP)

EMPIRE: Sandia plasma simulation code

- Relies heavily on capabilities in Sandia's Trilinos project
  - High-performance, portable shared memory primitives
  - Sparse distributed linear algebra
  - Distributed and shared memory solvers
  - Load-balancing, time-stepping
- Requires specialized algebraic multigrid (AMG) for Maxwell's equations

EMPIRE simulations have stringent solver performance requirements (10 solves/second)

EMPIRE – Sandia plasma simulation application

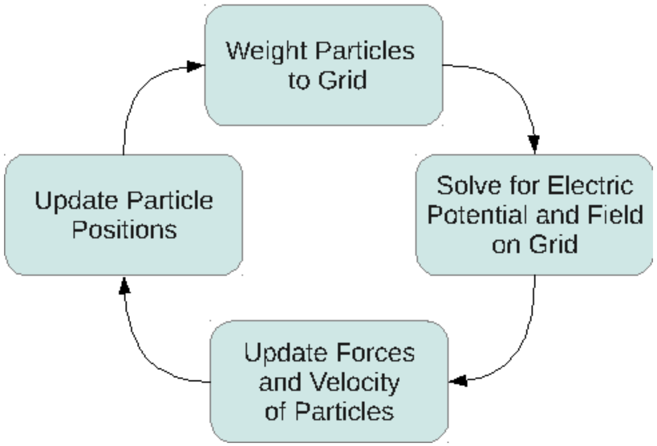
Plasma dynamics are described by Klimontovich equation

$$\frac{\partial f_i}{\partial t} + \frac{\vec{v}_i}{m} \cdot \nabla f_i + \frac{q_i}{m_i}\left(\vec{E}(x_i) + v_i \times \vec{B}(x_i)\right)\frac{\partial f_i}{\partial \vec{v}} = 0$$

for particles *i* with associated charge *q*, mass *m*, velocity v_i, and distribution f_i.

Particle movement coupled to electric field *E* and magnetic field *B* via Maxwell's equations.

EMPIRE uses operator-split time integration.



Bettencourt et al., *ASC ATDM Level 2 Milestone #6358: Assess Status of Next Generation Components and Physics Models in EMPIRE*, SAND2018-10100

Maxwell's Equations

$$\frac{\partial \vec{B}}{\partial t} = -\nabla \times \vec{E} \qquad \nabla \cdot (\epsilon \vec{E}) = \rho$$

$$\frac{\partial (\epsilon \vec{E})}{\partial t} = \nabla \times \mu^{-1} \vec{B} - \vec{J} \qquad \nabla \cdot \vec{B} = 0$$

$$\vec{n} \times \vec{E} = 0 \text{ on } \partial\Omega$$

electric permittivity ϵ and magnetic permeability μ.

Discretization of Maxwell's Equations

Discretized using nodal elements for $H^1(\Omega)$, Nedelec edge elements for $H(\mathrm{curl}, \Omega)$, and Nedelec face elements for $H(\mathrm{div}, \Omega)$.

Yields block system

$$\begin{pmatrix} \frac{1}{\Delta t}\boldsymbol{M}_B(1) & \boldsymbol{M}_B(1)\boldsymbol{C} \\ -\boldsymbol{C}^T\boldsymbol{M}_B(\mu^{-1}) & \frac{1}{\Delta t}\boldsymbol{M}_E(\varepsilon) \end{pmatrix}$$

where M_E and M_B are edge and face mass matrices, respectively

C is strong form of curl

Discretization of Maxwell's Equations (continued)

Block system can be factored

$$\begin{pmatrix} \frac{1}{\Delta t}\boldsymbol{M}_B(1) & \boldsymbol{M}_B(1)\boldsymbol{C} \\ -\boldsymbol{C}^T\boldsymbol{M}_B(\mu^{-1}) & \frac{1}{\Delta t}\boldsymbol{M}_E(\varepsilon) \end{pmatrix} = \begin{pmatrix} \frac{1}{\Delta t}\boldsymbol{M}_B(1) & \boldsymbol{0} \\ -\boldsymbol{C}^T\boldsymbol{M}_B(\mu^{-1}) & \boldsymbol{I} \end{pmatrix} \begin{pmatrix} \boldsymbol{I} & \Delta t\boldsymbol{C} \\ \boldsymbol{0} & \boldsymbol{S}_E \end{pmatrix}$$

where the Schur complement $\boldsymbol{S}_E$ is given by

$$\boldsymbol{S}_E = \frac{1}{\Delta t}\boldsymbol{M}_E(\varepsilon) + \Delta t\boldsymbol{C}^T\boldsymbol{M}_B(\mu^{-1})\boldsymbol{C} \qquad (1)$$

Challenge: $\boldsymbol{S}_E$ has large near nullspace (space of gradients)

Linear Solver Algorithm (1)

Augmenting $\boldsymbol{S}_E$ with grad-div term to reduce nullspace:

$$\overline{\boldsymbol{S}}_E = \frac{1}{\Delta t}\boldsymbol{M}_E(\varepsilon) + \Delta t\left\{\boldsymbol{C}^T\boldsymbol{M}_B(\mu^{-1})\boldsymbol{C} + \boldsymbol{M}_E(1)\boldsymbol{G}\boldsymbol{M}_\rho(\mu)^{-1}\boldsymbol{G}^T\boldsymbol{M}_E(1)\right\}$$

Solving (1) is equivalent to solving:

$$\begin{pmatrix} \overline{\boldsymbol{S}}_E & \frac{1}{\Delta t}\boldsymbol{M}_E(\varepsilon)\boldsymbol{G} \\ \frac{1}{\Delta t}\boldsymbol{G}^T\boldsymbol{M}_E(\varepsilon) & \frac{1}{\Delta t}\boldsymbol{G}^T\boldsymbol{M}_E(\varepsilon)\boldsymbol{G} \end{pmatrix}\begin{pmatrix} \tilde{\vec{c}}_E \\ \tilde{\vec{c}}_\rho \end{pmatrix} = \begin{pmatrix} \vec{F} \\ \boldsymbol{G}^T\vec{F} \end{pmatrix} \qquad (2)$$

where

$$\boldsymbol{S}_E\vec{c}_E = \vec{F}$$

$$\vec{E} = \tilde{\vec{c}}_E + \boldsymbol{G}\tilde{\vec{c}}_\rho$$

$$\tilde{\vec{c}}_E = \boldsymbol{M}_E(1)^{-1}\boldsymbol{C}\boldsymbol{M}_B(\mu^{-1})\tilde{\vec{c}}_B$$

Bochev et al., *An algebraic multigrid approach based on a compatible gauge reformulation of Maxwell's equations*, 2008.

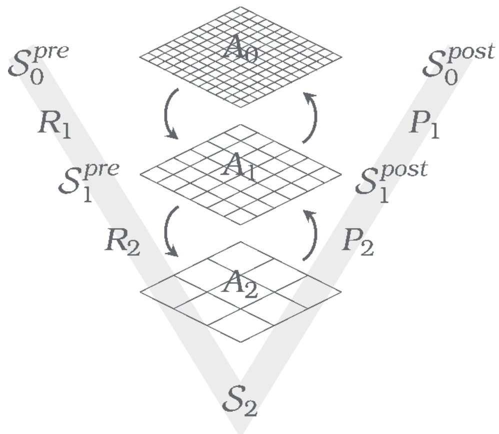
Linear Solver Algorithm (2)

Equation (2) can be solved with CG and Maxwell-specific algebraic multigrid preconditioner

- Block diagonal preconditioner
  - Off-diagonal coupling is ignored
- (1,1) block is edge-based vector Laplacian, requires special prolongator
- (2,2) block is node-based Laplacian

Multigrid: scalable solution method for linear systems arising from elliptic PDEs

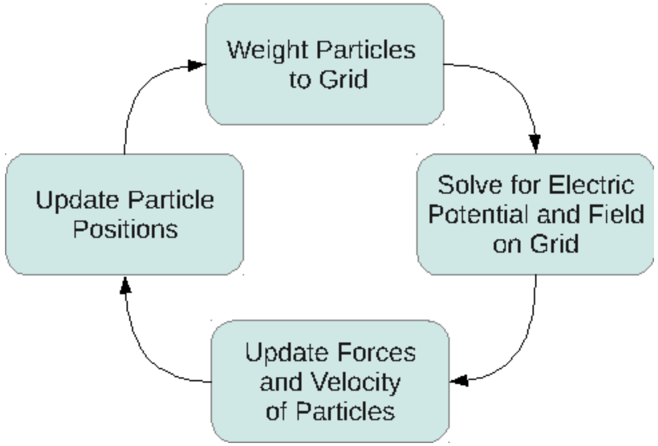
- Idea: capture error at multiple resolutions:
  - **Smoothing** reduces oscillatory error (high energy)
  - **Coarse grid correction** reduces smooth error (low energy)
- Algebraic multigrid (AMG)
- Preconditioner generates A_i's, R_i's, P_i's



Bochev et al., *An algebraic multigrid approach based on a compatible gauge reformulation of Maxwell's equations*, 2008.

Overall EMPIRE Solve

Recall that the Maxwell solve is part of overall EMPIRE solution process;



AMG preconditioner is set up only once, and then applied at each time step.

Efficiency of linear preconditioner apply is paramount.

Software

MueLu Multigrid Library

Unstructured algorithms

- AMG for Maxwell's equations
- classic smoothed aggregation (SA)
- non-symmetric AMG

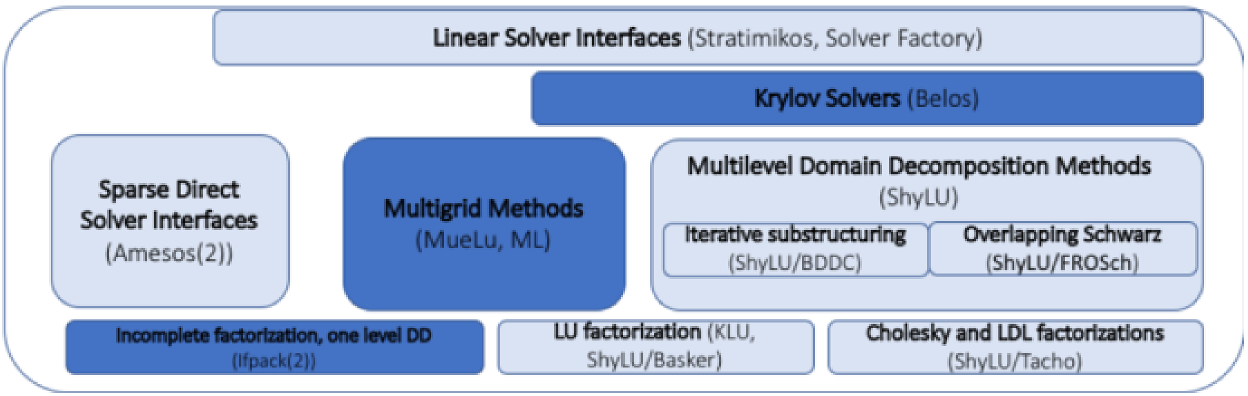
Structured Algorithms

- semi-coarsening AMG
- geometric MG
- structured-grid aggregation-based MG

Leverages many other Trilinos scientific libraries

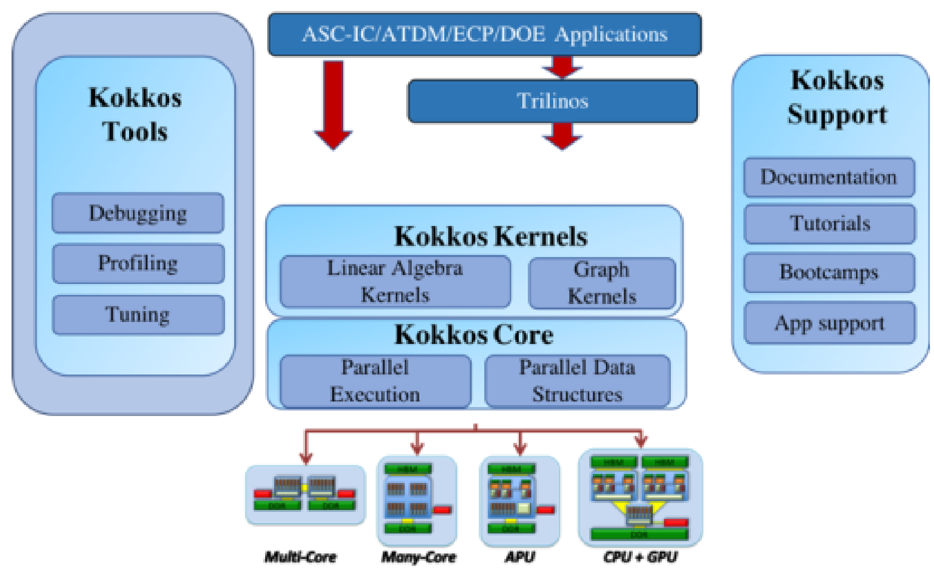
- Shared memory parallelism from Kokkos → architecture portability
- Sparse distributed linear algebra: Tpetra
- Distributed smoothers: Ifpack2
- Shared memory smoothers, SpGEMM, distance-2 coloring: Kokkos-Kernels
- Load balancing: Zoltan2
- Direct Solvers: Amesos2

Trilinos Project



github.com/trilinos/Trilinos

github.com/kokkos



Cross-Architecture Algorithm components

Kokkos ecosystem underpins many solver aspects

- Multigrid polynomial smoothers
  - Sparse matrix-vector multiply and residual kernels
- Grid transfers
- Coarse matrix construction
  - Sparse matrix-matrix multiply
- Krylov solvers

General observation

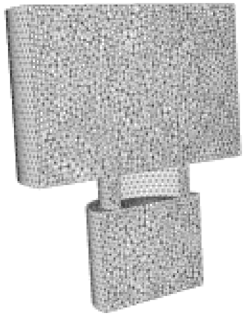
- Replacing parallel primitives isn't necessarily sufficient
  - New architectures may require new algorithms

Numerical Results

Large Scale Problem Descriptions

Simple BDot Cavity

- Simplified physics
- Preloaded particles.
- Run for nominal 100 time-steps to gather metrics.



Mesh	Elements	Nodes	Edges	Particles
R0	337k	60.4k	406k	16M
R1	2.68M	462k	3.18M	128M
R2	20.7M	3.51M	24.4M	1.0B
R3	166M	27.9M	195M	8.2B
R4	1.33B	223M	1.56B	66B

Generic Cavity

- Complex geometry
- Preloaded particles for scaling studies
- Run for nominal 100 time-steps for scaling studies.

Mesh	Elements	Nodes	Edges	Particles*
R0	3.7M	660k	4.4M	360M
R1	25M	4.4M	30M	2.4B
R2	200M	32M	240M	19B
R3	1.6B	270M	1.9B	160B

*Scaling runs

Platforms

Sierra (ATS-2)

- Power9 CPUs, Nvidia V100 GPUs
- Infiniband

Astra (Vanguard)

- ARM Cavium Thunder-X2 processors
- 2592 nodes / 145,152 cores
- Infiniband

Trinity (ATS-1)

- Intel Haswell / Intel Phi
- 19,420 nodes
- Cray Aries

Linear Solver Details

Recall the system that we are solving:

$$\begin{pmatrix} \frac{1}{\Delta t}\boldsymbol{M}_B(1) & \boldsymbol{M}_B(1)\boldsymbol{C} \\ -\boldsymbol{C}^T\boldsymbol{M}_B(\mu^{-1}) & \frac{1}{\Delta t}\boldsymbol{M}_E(\varepsilon) \end{pmatrix} = \begin{pmatrix} \frac{1}{\Delta t}\boldsymbol{M}_B(1) & \boldsymbol{0} \\ -\boldsymbol{C}^T\boldsymbol{M}_B(\mu^{-1}) & \boldsymbol{I} \end{pmatrix}\begin{pmatrix} \boldsymbol{I} & \Delta t\boldsymbol{C} \\ \boldsymbol{0} & \boldsymbol{S}_E \end{pmatrix}$$

$\boldsymbol{S}_E$ solved with Conjugate Gradient + Maxwell AMG

Maxwell AMG preconditioner is set up only once, applied over many timesteps

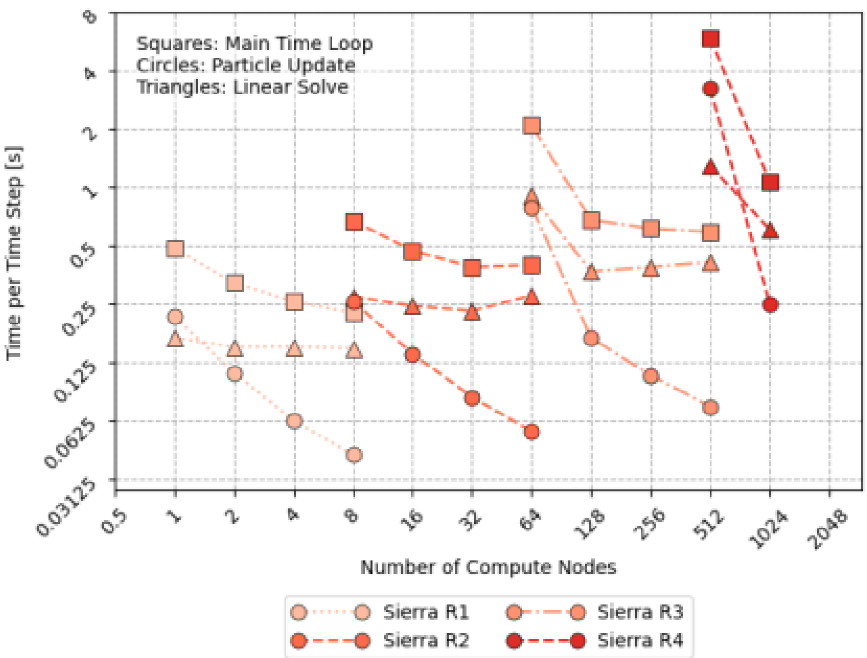
Fine grid smoother: degree 4 Chebyshev polynomial

We will use 2x2 block diagonal (additive) AMG variant

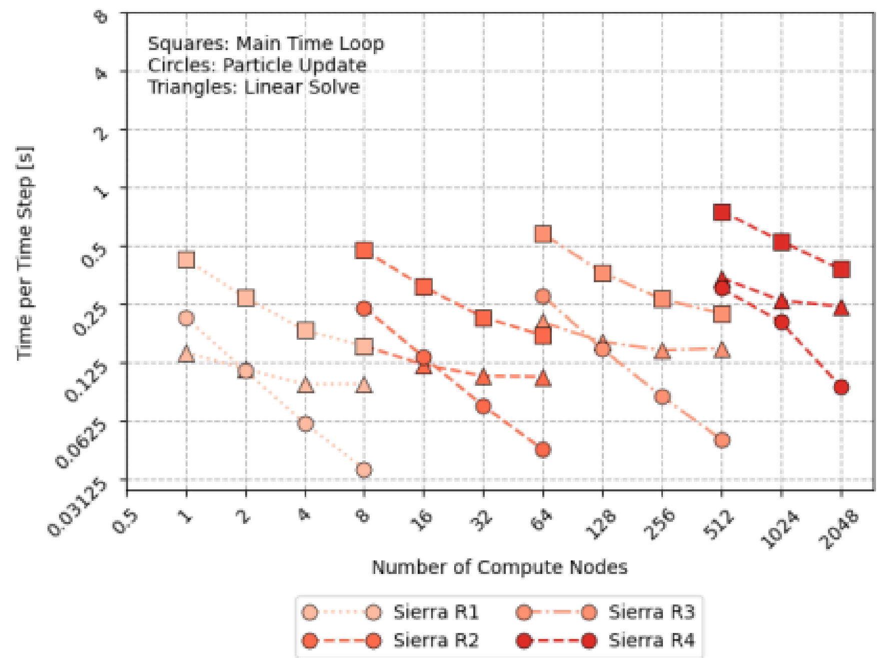
- (1,1) block: single level, degree 6 Chebyshev
- (2,2) block: single level, degree 6 Chebyshev

Sierra Supercomputer performance improvements (BDot cavity reference problem)

December 2019 results



August 2020 results



▽ Linear solver did not weak scale

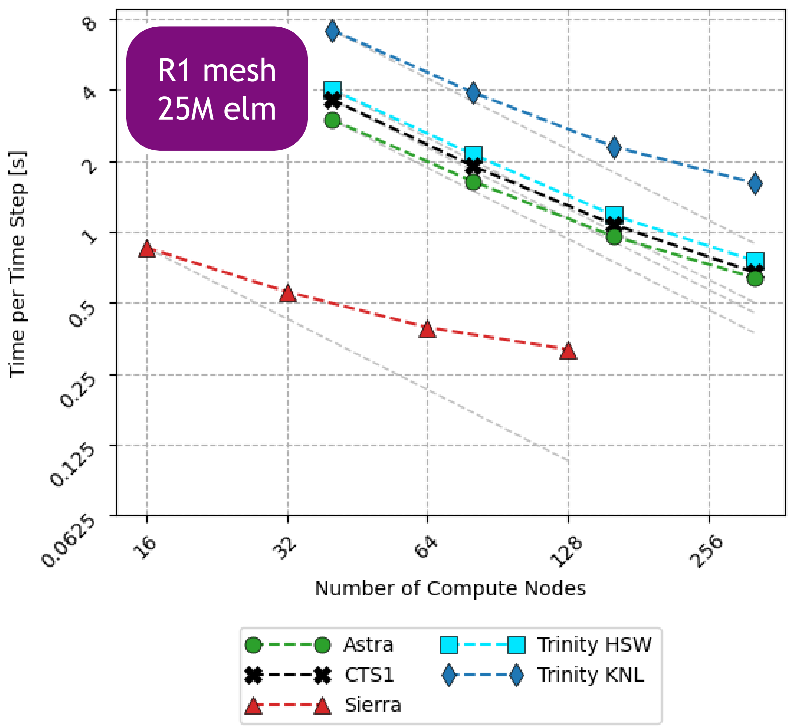
▽ Particle update showed strong scaling issues

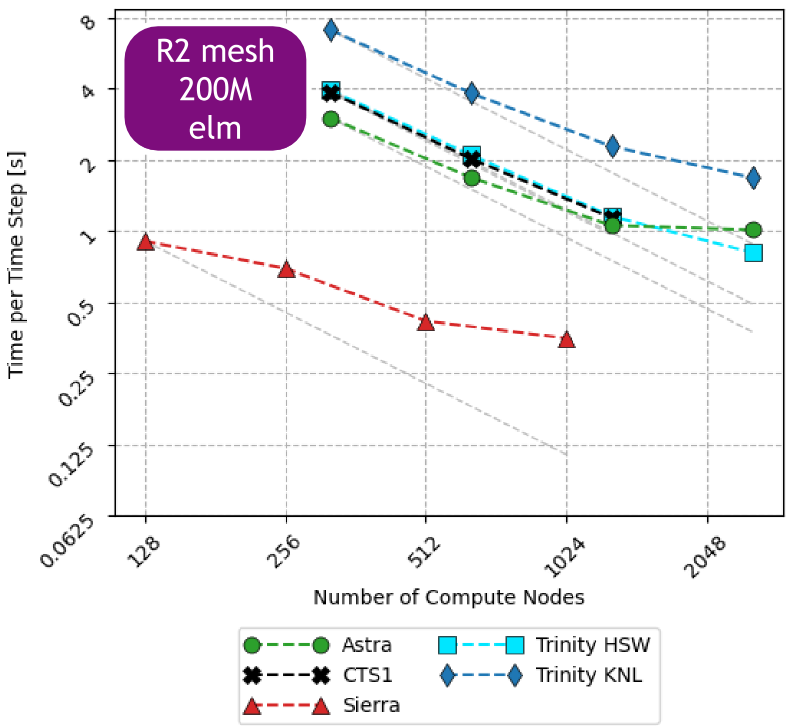
△ Solver performance and scaling improved

△ Particle performance and scaling improved

Performance results for Generic Cavity

Cross-platform strong scaling results

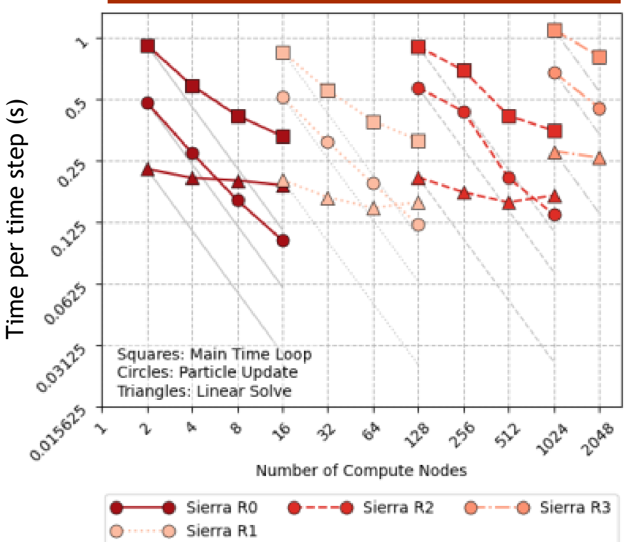




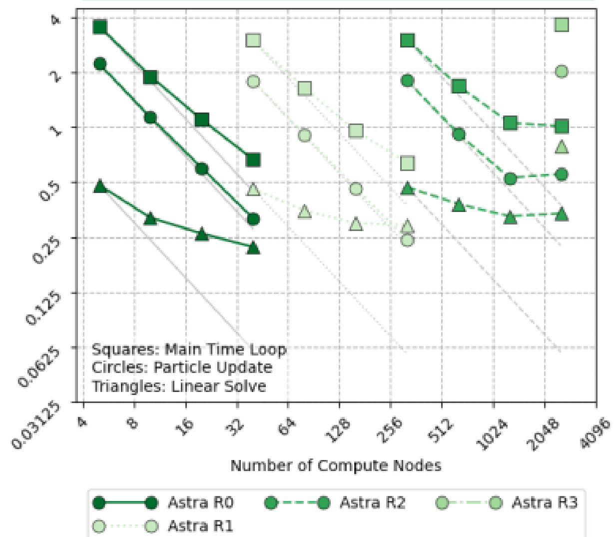
Performance results for Generic Cavity

Single platform scaling results

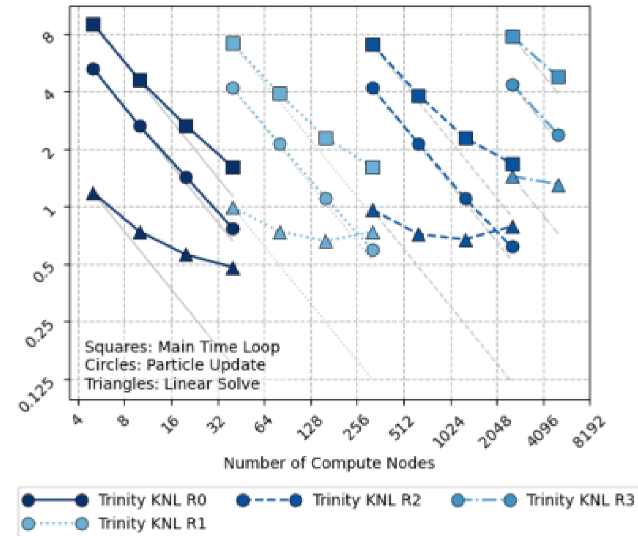
Sierra
4 GPU/socket; 1 MPI/GPU
Up to 2048 nodes (47% of total)



Astra
2 MPI/socket; 9 threads/MPI
Up to 2560 nodes (99% of total)



Trinity/KNL
4 MPI/socket; 16 threads/MPI
Up to 5120 nodes (52% of total)



Concluding Remarks / Future Work

Architecture portable solvers play key role in field solve in PIC simulations

Future Work:

- Portability
  - Continue removing reliance on UVM
  - Adapt solvers to new architectures (e.g., AMD)
- Algorithms
  - Develop solvers for high order E&M
  - Reuse/amortize solver setup across solves

Related talks/minisymposia

- Wedn., 7-July, "Properties of GMRES with Iterative Refinement on GPUs", J. Loe et al.
- Wedn. 7-July, Developing Scientific Codes for Predictive Simulations on Massively Parallel Heterogeneous Computing Platforms: Integrating Extreme-Scale Computation, Data Analysis and Visualization, part 1
- Friday 9-July, Developing Scientific Codes for Predictive Simulations on Massively Parallel Heterogeneous Computing Platforms: Integrating Extreme-Scale Computation, Data Analysis and Visualization, part 2