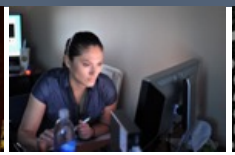
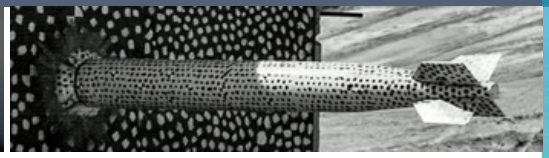
SAND2023-11790PE UUR

Trilinos Core Product Area Update


PRESENTED BY

Roger Pawlowski

Current Package Owners: R. Bartlett, L. Berger-Vergiat, E. Boman, C. Glusa, S. Rajamanickam, C. Siefert, G. Sjaardema, C. Trott

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.

Trilinos Core Product Area

- Teuchos: Common Trilinos utilities
- Tpetra: Parallel sparse linear algebra
- Xpetra: Abstractions to switch between Epetra and Tpetra
- PyTrilinos(2): Python wrappers
- Pamgen: Inline mesh generation utility
- RTOp: Reduction/transformation operators
- Thyra: Abstraction layer for Trilinos

- Snapshotted Packages:
  - Kokkos: Performance portability
  - KokkosKernels: Performance portable linear algebra
  - SEACAS: Finite Element tools for Exodus database format

- Deprecated Packages:
  - Epetra
  - EpetraExt
  - Isorropia
  - TriUtils

Teuchos

- Mostly in maintenance mode

- Improved YAML Parser Support [TRILINOS-192]
  - Current: Homegrown parser that has incomplete standard support
    - Application requests: tabs, ragged arrays, arrays split across lines, unicode in comments, …
    - Not clear the internal design is capable of supporting requests
  - Options: (1) fix native parser, (2) yaml-cpp, (3) libyaml, …
  - Leadership team: long term prefer to offload yaml parsing to an external library, Trilinos does not need to own this.
  - Transition: identify cpp yaml parser library and add support along side the internal one. Allow for both to exist during transition (configure flag to switch).
  - Opinions/preferences?

- Trilinos Leadership: discussions on what we can clean up and remove
  - Keep common look and feel (e.g. ParameterLists, scalar_traits, …)
  - New c++ standards (e.g. Teuchos::any → std:any)
  - RCPs/memory management tools (RCP → shared_ptr, Array→std::vector)?
  - Test harness (Teuchos → gtest)?

Tpetra

- FY23 focused on: Performance, Performance Testing/Monitoring, software quality improvements and Epetra→Tpetra transition

- Talks this week (Wed. afternoon):
  - *GPU H2D/D2H + Fence Tooling* - Tools you can use to help identify (and regression test) H2D/D2H transfers and fences in your Trilinos-based code. Will include profiling results from some apps / app proxies.
  - *Epetra to Tpetra Transition* - How to move your code from Epetra to Tpetra, in two parts. First, transitioning to Tpetra without Kokkos. Second, how to use Kokkos w/ Tpetra using FE assembly as an example.
  - *Performance Profiling* - A peak into nightly performance & memory app proxy testing done by the Tpetra team. Should your app be taking part too?

- Planned for FY24
  - Application transition support: Epetra → Tpetra
  - Cleanup unnecessary D2H and H2D transfers

Epetra Deprecation Plan

- Deprecation of Epetra from Trilinos (FY24)
  - All Trilinos packages compile and function without Epetra
    - Nightly Testing without Epetra (done)
  - All “needed” Epetra Testing has equivalent Tpetra Testing
  - Packages consult with Stakeholders to determine any missing Tpetra functionality
    - Try to assess performance impacts
  - Deadline end of FY24 (Sep. 2024)
    - Epetra still available and tested during FY24

- Deprecation of Epetra from Stakeholder applications (FY25)
  - Applications transition to Tpetra
  - Packages handle any new issues and performance problems
  - Deadline end of FY25 (Sep. 2025)

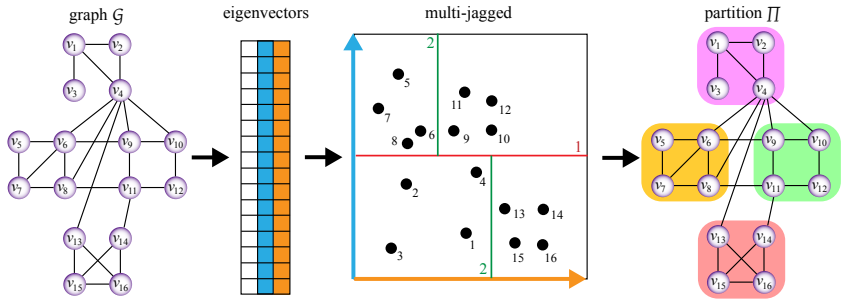
- Beginning of FY26 (Oct. 2025) Epetra stack archival to separate repo.

Parallel partitioning in Zoltan2

Two new graph partitioners in (coming to) Zoltan2:

- Sphynx: Spectral partitioner, multi-GPU
  - Randomized eigensolver speeds up initial version by up to 50X
- Jet: Multilevel, Kokkos-based partitioner (CPU and GPU)
  - High quality. Beats Metis by 6-10% in edge cuts. Coming soon.
  - Currently limited to single GPU, multi-GPU in progress.

Sphynx example: k=4 parts



[1] Acer, Boman, Glusa, Rajamanickam, "Sphynx: A Parallel Multi-GPU Partitioner for Distributed Memory", Parallel Computing 106, 2021

[2] Gilbert, Madduri, Boman, Rajamanickam: "Jet: Multilevel Partitioning on GPUs", SISC, to appear.

Other Packages

- PyTrilinos2 (new package)
  - Auto-generation of Python interface using PyBind11 and binder
  - See “Disc. and Analysis Update” at 11:45am today for an overview from K. Liegeois!
- PyTrilinos
  - Plan to deprecate and remove original implementation based on SWIG wrappers (no timeline at this point)
- RTOp
  - Maintenance mode
- Thyra
  - Maintenance mode
- Xpetra
  - Maintenance mode
  - Leadership: Is this a potential candidate for deprecation with Epetra removal?

Kokkos and Kokkos-Kernels

- POC:
  - Kokkos: Christian Trott
  - Kokkos-Kernels: Siva Rajamanickam, Luc Berger-Vergiat

- Kokkos drives C++ standards requirements
  - Current minimum is C++17
  - Require C++20 in mid-2025

- Can now build Trilinos against an external install of Kokkos!
  - Trilinos Spack builds do this by default (see Ross B. talk on Thursday)

- Releases:
  - Kokkos 4.2 in November 2023
    - Kokkos CHANGELOG: https://github.com/kokkos/kokkos/issues/6197

FY24 Kokkos-Kernels Planned Work

Algortihms

- Block-ILU(k) variant
  - Fill based on block graph
  - Integration with Ifpack2
- LAPACK select algorithms implementation (LU, SVD, QR)
  - Add new library component
  - Include cuSOLVER, rocSOLVER, MKL and Magma TPLs
- Improve BDF features:
  - Numerical differentiation Jacobian
  - Backtracking line search
- Batched ODE solvers
  - Reduce branch divergence on GPU
  - Promote vectorization on CPU
  - Potentially complicated for BDF, easier for RK algorithms

Library

- SYCL backend: improve support and performance once Aurora comes online
- Improve integration with Trilinos and PETSc
- Establish automated performance testing
- Improve interface to enable auto-tunning

SEACAS Improvements

- Aprepro:
  - Full precision output -- shortest decimal representation with round-trip guarantee
  - Loop syntax improved {loop(ncount, index, initial, increment)}
  - Easier echo on/off via `{{a = b}}`. The expression will not be echoed.
  - Some fundamental physical constants are now predefined.
  - See the documentation for more details on each of these.
- Exodus:
  - Get/Put variables over multiple timesteps
  - Enhanced Field /Discontinuous Galerkin – In progress
  - Improved Python interface (more pythonic, more complete)
  - Improved Assembly handling
- IOSS:
  - TextMesh, Null, NullExodus database types
  - Robustness Improvements
  - Zero-Copy fields (primarily for Catalyst)
- Other
  - Robustness improvements (static analyzers, …)
  - Slice (yet another decomposition method)
  - Io_shell has toleranced comparison option
  - Io_modify for defining/modifying assemblies; other options.