SAND2022-14675C

Sandia National Laboratories

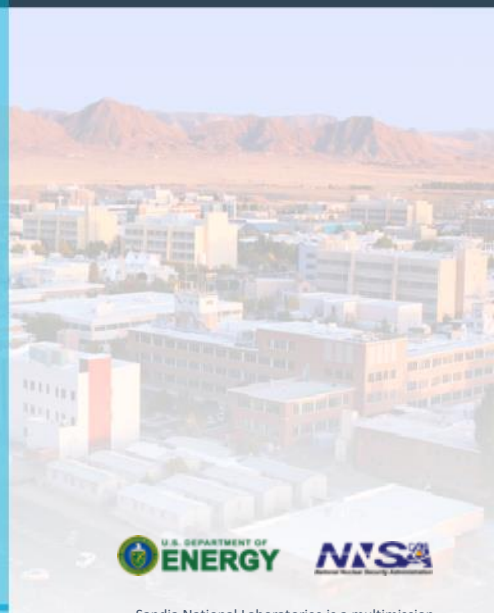
Tpetra in FY23





PRESENTED BY

Chris Siefert, Tpetra /Performance Team Lead



U.S. DEPARTMENT OF ENERGY | NNSA

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and 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.

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.

Tpetra in FY22

Removal of UVM requirement

- UVM-free Tpetra stack now tested in PR testing.

Removal of deprecated code in 13.4

- Over 27k lines removed by the Tpetra team.

New platform support

- AMD/HIP (not PR tested).
- Intel/SYCL (not regularly tested).

Asynchronous Import/Export

- Motivation
  - Import/Export transfer data from one distributed object (`Tpetra::DistObject`) to another
  - Let's say you have many MultiVectors to do import on …
  - What if you want to overlap communication?
    - Launch sends for multiple DistObjects simultaneously
    - Launch sends and do some other computation while you wait
- Synchronous API
  - Do the complete import, don't return until it's finished: `DistObject::doImport`
- New asynchronous API
  - Pack data and kick off sends: `DistObject::beginImport`
  - (Optionally) check if data has arrived and is ready to unpack: `DistObject::transferArrived`
  - Unpack and combine data: `DistObject::endImport`
- Backend improvements mean each DistObject handles communication separately
  - BUT, can still share the same communication plan from the importer (expensive to create)

Lead developer: Timothy Smith

Prototype: On-node graph assembly

- For on-node matrix assembly, we've had an interface for quite some time…
  - Grab the Kokkos::SparseCrsMatrix and work on that directly.

- But how do you assembly a *Graph* on-node?
  - For many apps, host-assembly suffices --- the connectivity never changes.
  - But some apps have Graphs that change over time.

- Brian Kelley has been working on a FEM-centric prototype for graph assembly:

```
RCP<CrsGraph> Tpetra::assembleFEGraph(
                          RCP<Map> rowMap,
 View<GO**, Node::memory_space> ownedElements,
View<GO**, Node::memory_space> ghostElements);
```

- Still in development in FY23.

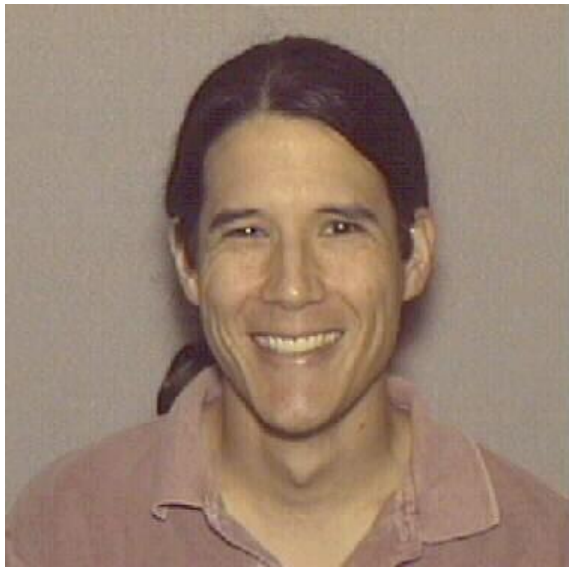
Lead developer: Brian Kelley

Improved BlockCrsMatrix Support

- Tpetra::BlockCrsMatrix was designed to support fixed-sized, small, blocks, e.g., 5x5.
- Uses a CrsGraph on *nodes* (groups of dofs) for the blocked problem --- less pointer chasing than CrsGraph for each individual dof.
- New features
  - Transpose operation.
  - Sparse matrix-matrix multiplication.
- Enables blocks-through-the-whole-hierarchy in certain MueLu code-paths.

Lead developer: Conrad Clevenger

Faces you might see at our meetings



Tpetra FY23: Performance Testing

Emphasis on performance testing over code development

- Already have regular app tests (Sierra/TF, EMPIRE).
- Platforms: CTS1, ATS2, VAN1, ORNL/Crusher.
- Reviewed by humans every Tuesday.
- Jonathan will discuss this more on developer day.

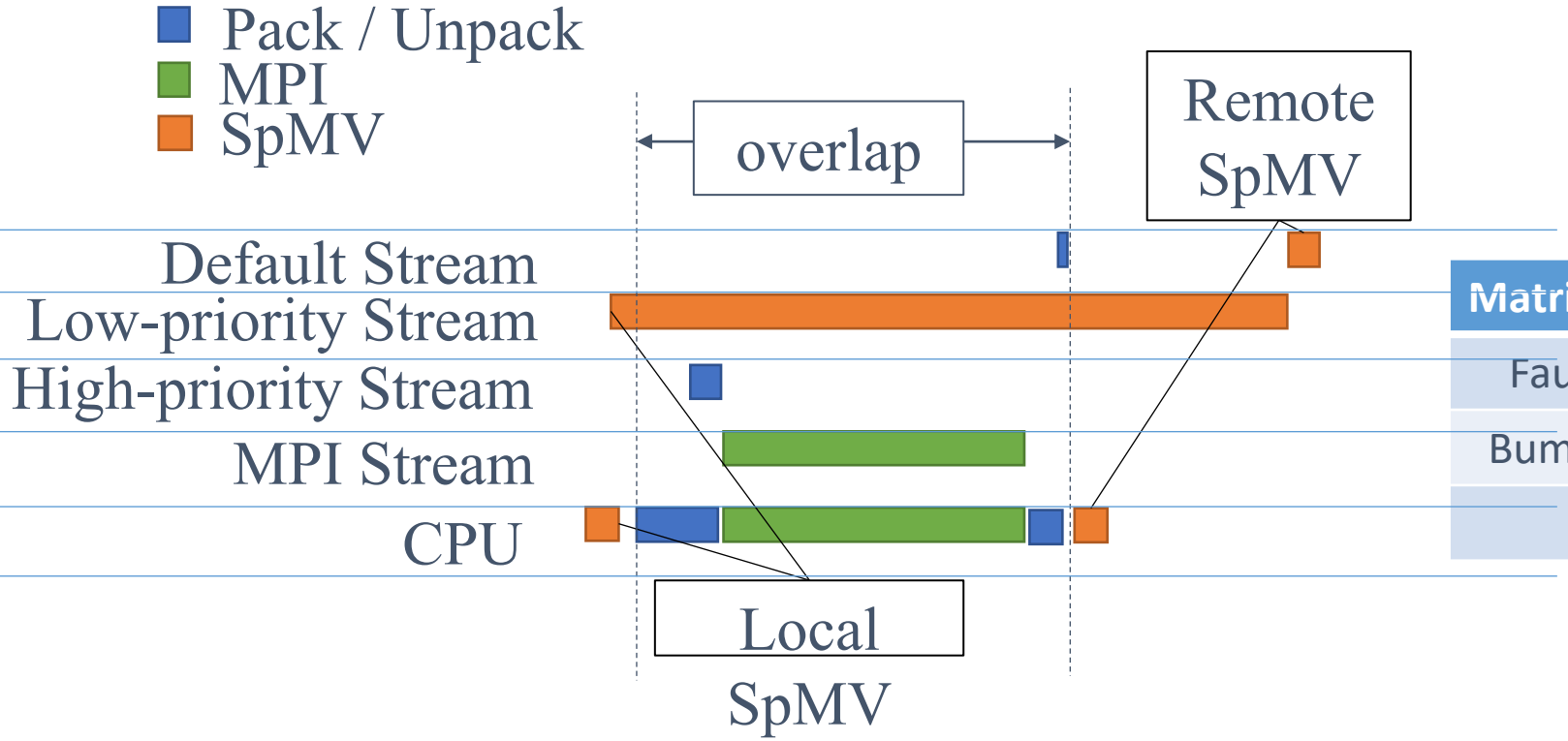
Goal: Add performance testing for more apps (Xyce,SD,???).

Tpetra FY23: Code Cleanup

Reducing memory high-water in the boundary exchange (as identified by Sierra/SD).

Time permitting: Refactor the SpGEMM code and push some code to KokkosKernels.

Tpetra FY23: Comp/Comm Overlap in GPU



Matrix	Config	Speedup.
Fault_639	Vortex, 1n 4r	1.16x
Bump_2311	Vortex, 2n 8r	0.9x
SM	Vortex, 8n 32r	1.59x

- Preliminary implementation and evaluation
- Expected to be opt-in behind a behavior (not always beneficial)
- Other common operations may benefit from similar changes

Lead developer: Carl Pearson

Questions?