SAND2021-4666C

kokkos

Kokkos Tools Update

Business Sensitive Information

David Poliakoff, Amy Powell, Jonathan Madsen

EXASCALE COMPUTING PROJECT

Sandia National Laboratories

Argonne NATIONAL LABORATORY

OAK RIDGE National Laboratory

BERKELEY LAB Lawrence Berkeley National Laboratory

Los Alamos NATIONAL LABORATORY EST. 1943

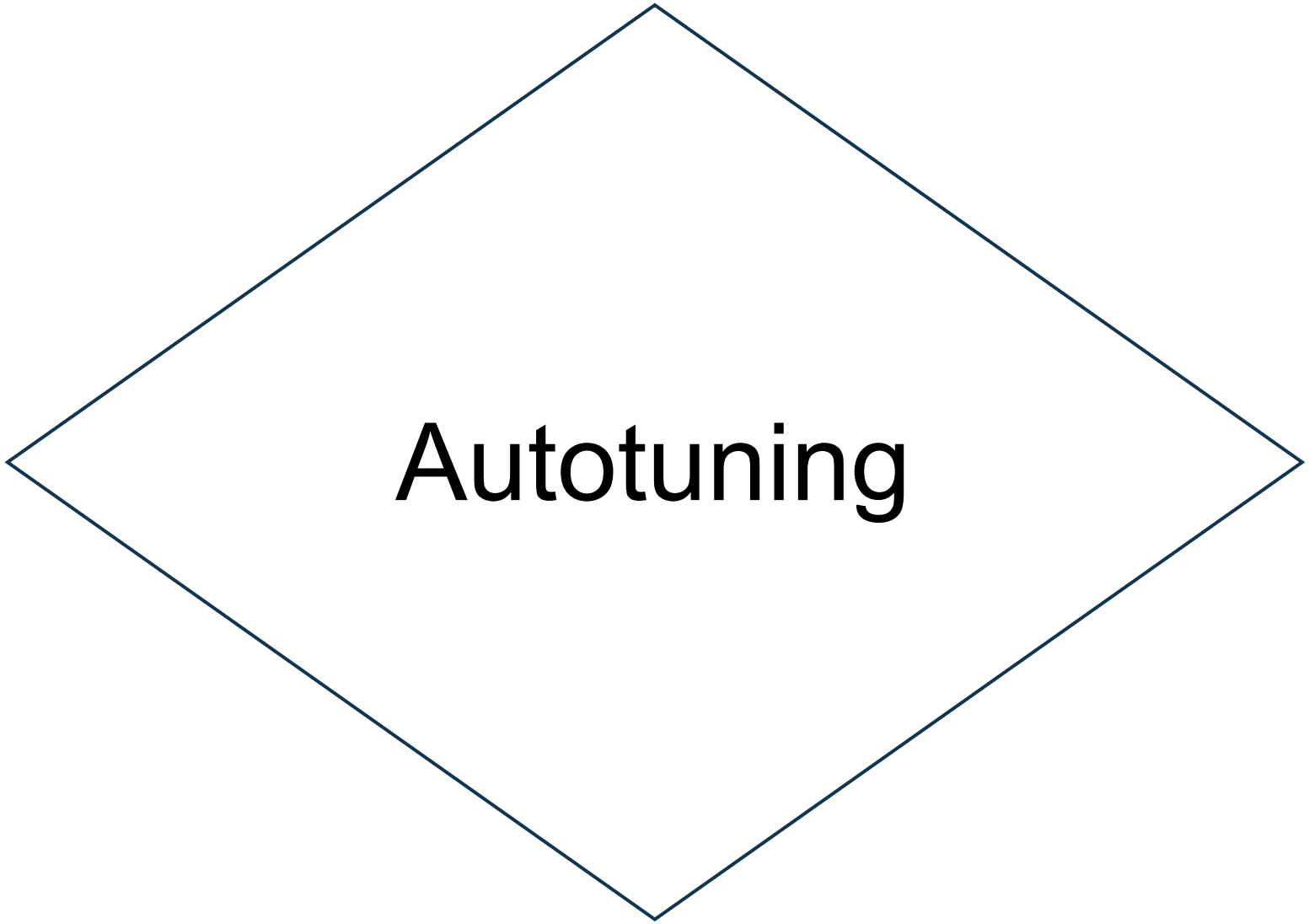
NNSA National Nuclear Security Administration

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-NA-0003525.

U.S. DEPARTMENT OF ENERGY | Office of Science

Another collaborative year!

Community project through and through


Autotuning

Why autotune?

The last 10-15% of performance in a Kokkos app comes from setting a few tuning knobs. These need to be maintained per:

Hardware: Intel, AMD, and NVIDIA GPU

Programming models: Serial, OpenMP, OpenMPTarget, CUDA, HIP, SYCL, Threads, HPX

Compilers: NVCC, Clang, GCC, vendor clang variants

How do you feel about maintaining that many heuristics?

Heuristics aren't feasible moving forward

Requirements: what can't we do?

- Recall from Profiling:
  - No recompilation
  - Applications can't *fail* if tools aren't available
  - No third-party dependencies in Kokkos
- Good news: there are many good tuning technologies in ECP we can use
- Bad news: there are *too many* good tuning technologies in ECP to pick one
- Answer: abstraction through a callback interface

Need to support a variety of tools, ECP has *depth* in this area

Based on our original tuner: Christian trott

- How does the Trott Tuner think about things like sparse matrix vector product?
  - What do I need to tune?
    - An implementation
    - Team sizes for my team policies
  - What might effect my choice?
    - Number of rows in the matrix
    - Sparsity
    - Backend I'm using
  - What options are valid?
    - Maybe a set, maybe a range?
  - Our autotuners need the information Christian has, provided formally

Need to support a variety of tools, ECP has *depth* in this area

App workflow: typical

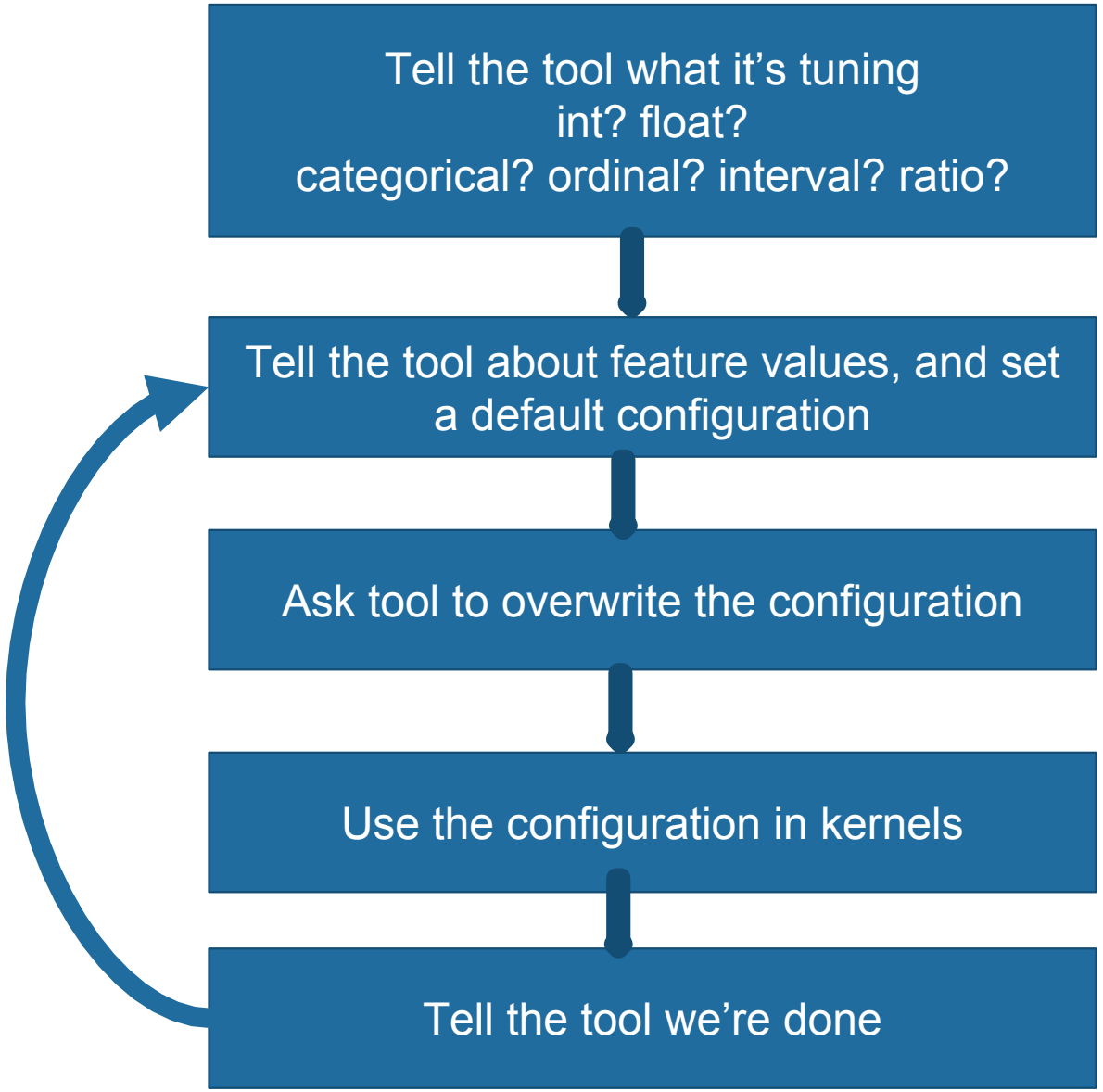
```
Kokkos::TeamPolicy<> policy(number_of_rows,
  Kokkos::AUTO,
  Kokkos::AUTO);

Kokkos::parallel_for(policy, /** ... */);
```

Absolute genius

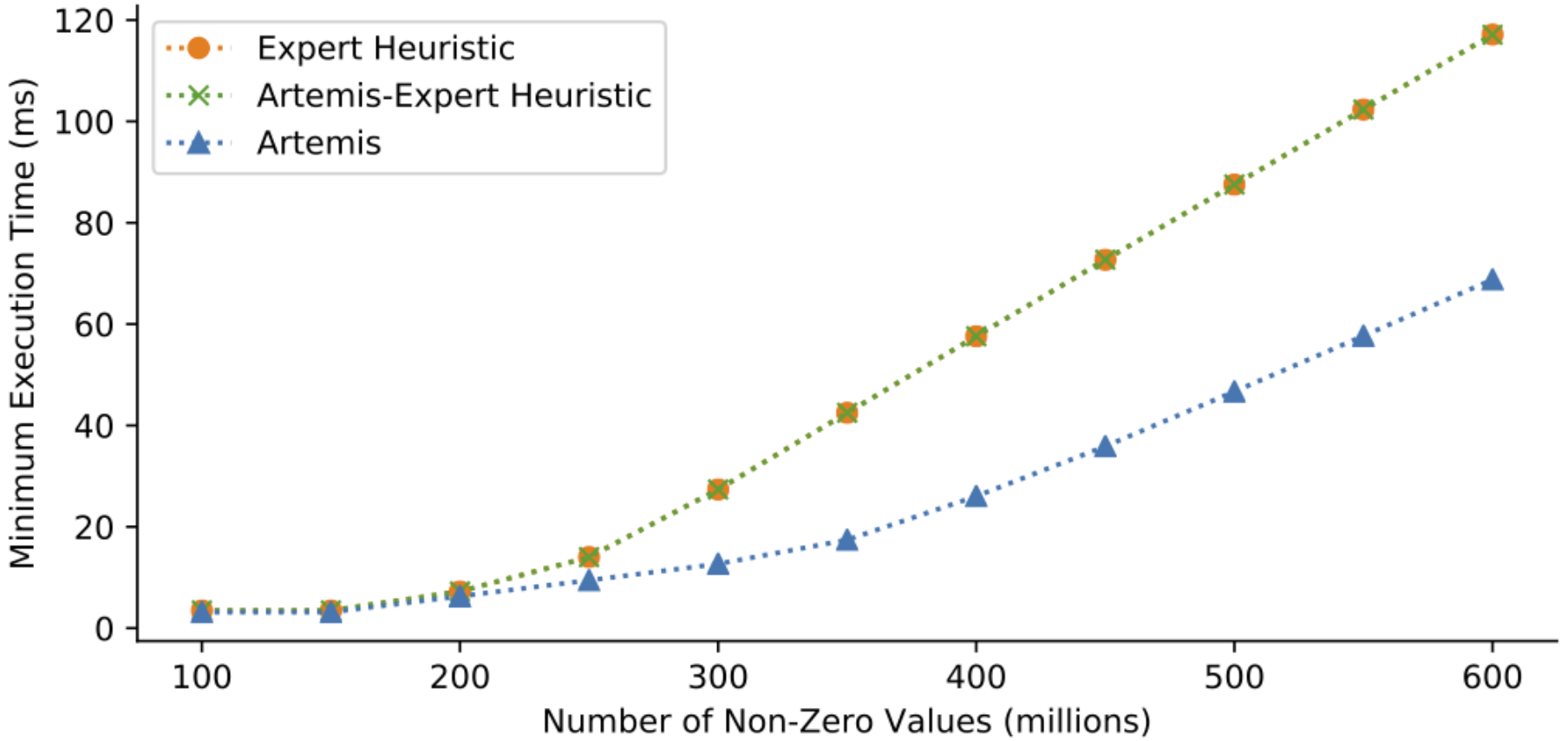
In most cases, code changes very little

App workflow: advanced



While app not done

Autotuning SPMV



Graph courtesy of Artemis team from LLNL and UO

Speedups on hand-tuned kernels. Reproduced with multiple tools


Performance
Tracking

Performance tracking: confident portable development

- Recently: CUDA performance on NVIDIA **priority one drop everything**
- Now: HIP or OpenMPTarget performance on AMD **priority one drop everything**
- Did your changes for HIP hurt your CUDA performance?
- Did *our* changes to HIP hurt your HIP performance?

- Solution survey:
  - SPOT
  - Watchr
  - LDMS
  - Dana
- Kokkos tools insulate you from betting your code on any one of these!

During frenetic development performance tracking *critical*

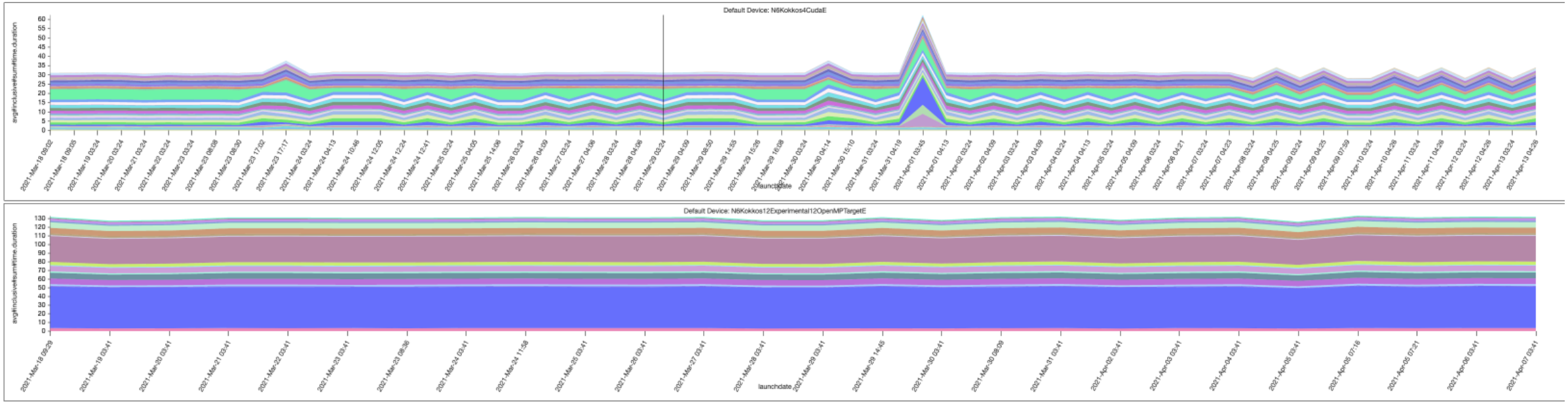
Code changes

```
Kokkos::Tools::declareMetadata("input_deck",
  input_deck);
Kokkos::Tools::declareMetadata("num_ranks",
  std::to_string(num_ranks));
Kokkos::Tools::pushRegion("region");
Kokkos::Tools::popRegion();
```

As always: easy to integrate!

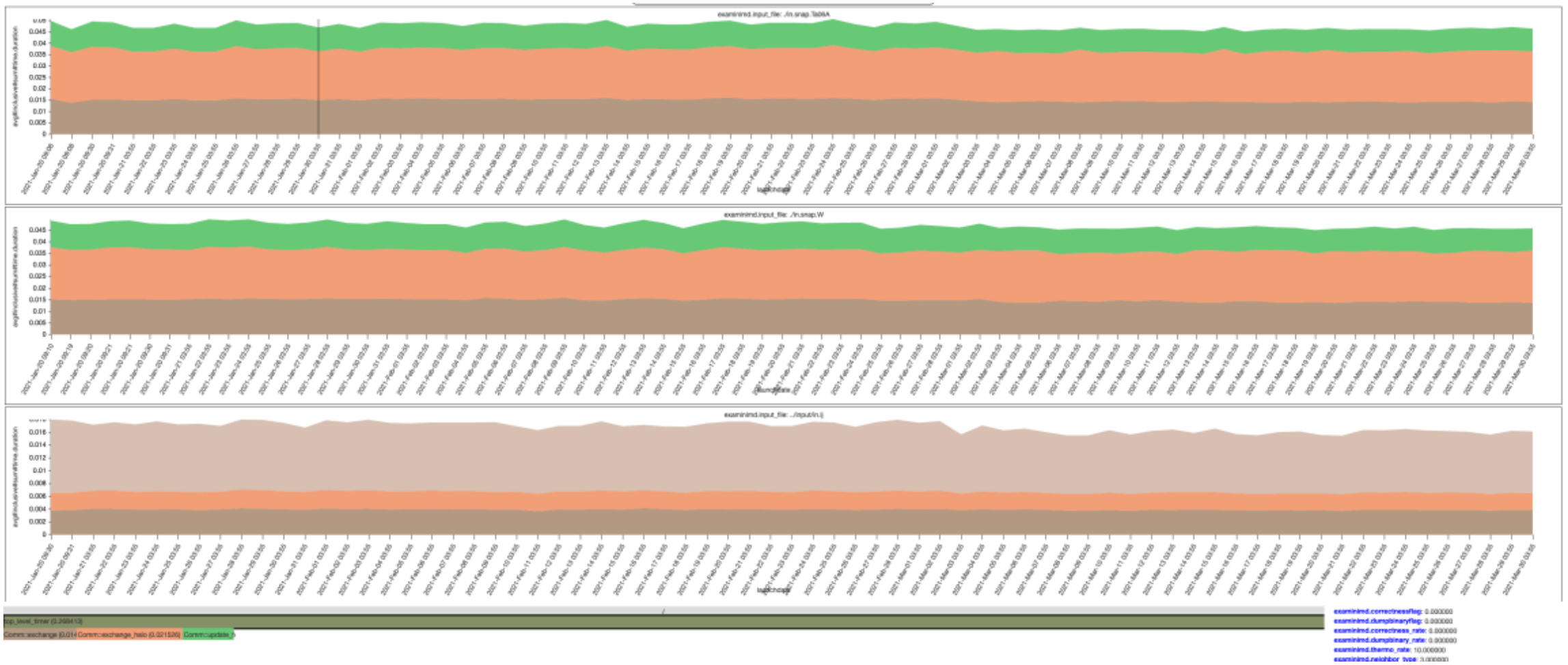
RAJA Perfsuite testing

- Adapted Rich Hornung's RAJA Perfsuite to test whether Kokkos has introduced new complexity to make optimization impossible



Graphs don't slope up: we get to keep our jobs!

ExaminiMD



We didn’t ruin our customers’ days either!

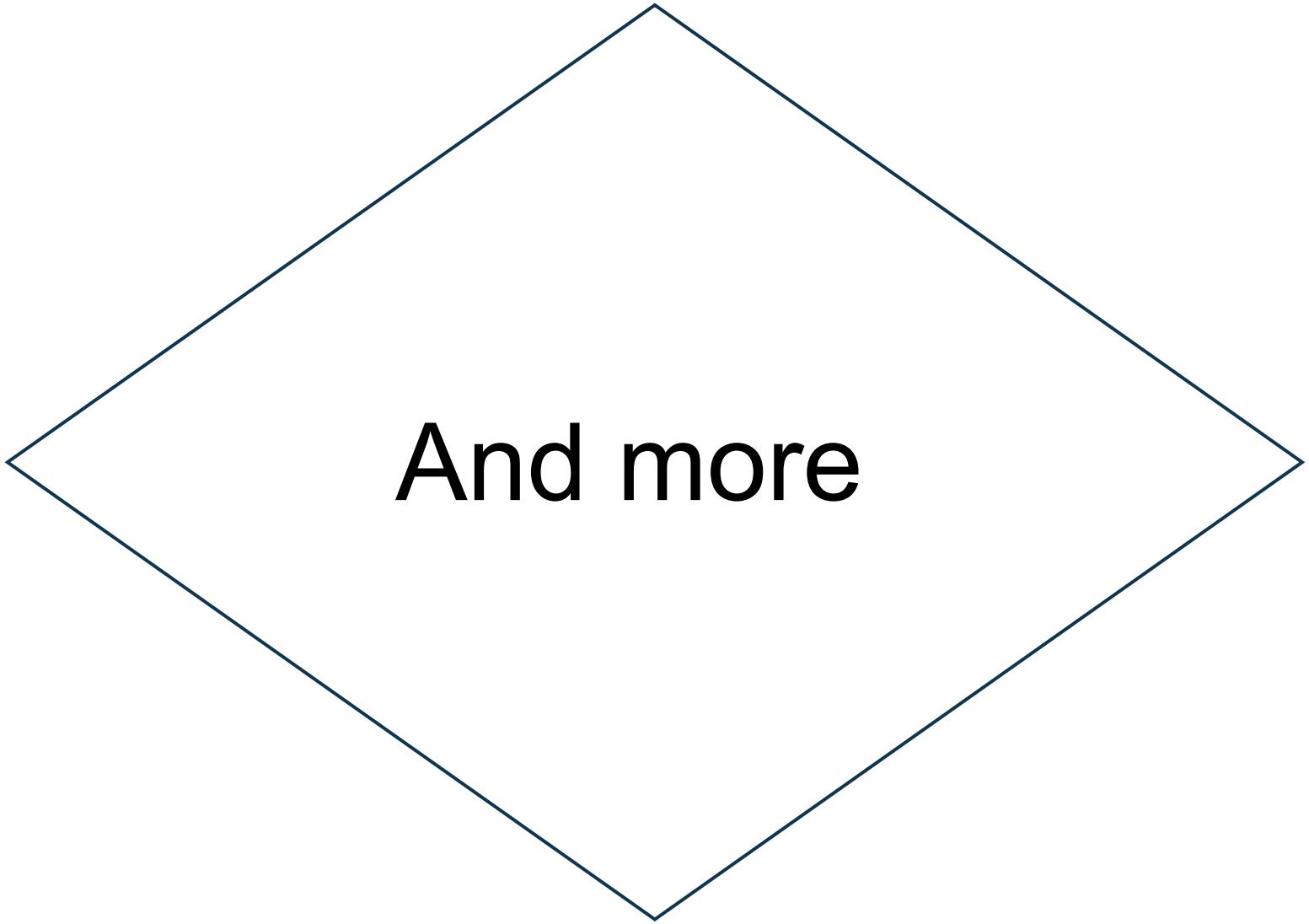
Spot info

- Option one: time machine, take the tutorial they gave Monday
- Option two: email David Boehme (boehme3@llnl.gov) and ask for the recording

- Working on hosting the web infrastructure at ORNL and NERSC

- Also in progress: Watchr integration (Jenkins plugin based performance tracking), LDMS streams

Moving to production


And more

The laundry list

- Improvements to allow tools without adding Kokkos fences
- Tools to track UVM data movement
- Updates in Kokkos Clang-Tidy
- Additional autotuning tools (Labrador and CConfigSpace and Apex)
- DualView callbacks to track DualView actions
- Power monitoring with Variorum
- Improved TAU support
- Timemory support
- Godbolt/Compiler Explorer instances
- **kokkosteam.slack.com**
- **dzpolia@sandia.gov**