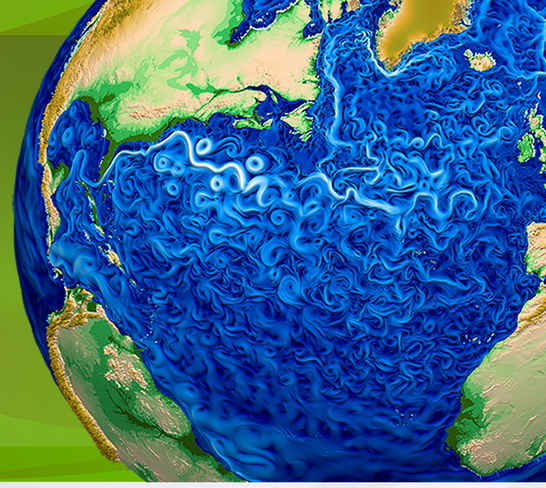
P: C++/Kokkos Refactor of the HOMME Dycore: CMDV Software

Luca Bertagna, Michael Deakin, Oksana Guba
Dan Sunderland, Andy Salinger, Irina Tezaur



Background and Plan

CMDV Software Modernization Project

Goal: To improve the state of ACME software to accelerate scientific discovery

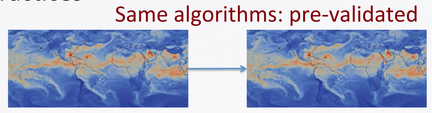
Proposal themes:

1. To move DOE climate modeling towards decision support
2. To improve agility for ongoing architecture disruption
3. To fully leverage DOE computational science expertise

Refactor of HOMME Spectral Element Dycore to use C++/Kokkos/Trilino

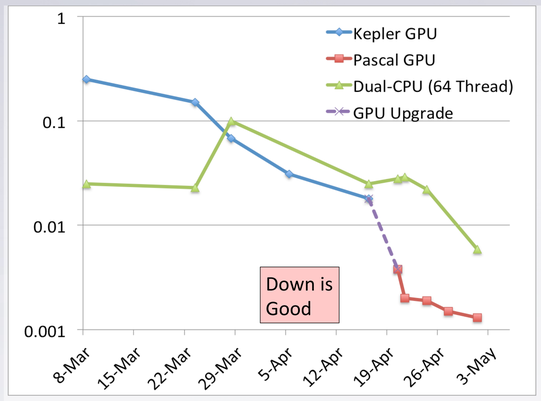
- Develop a single code base using C++ & Kokkos that performs well on CPU, Phi, and GPU architectures
- Leverage Kokkos development for good performance on future architectures
- Demonstrate Rapid Development by using modern software practices
- Entrain talented CS-types to think about ACME performance
- Access advanced algorithms via Trilinos

Same algorithms: pre-validated

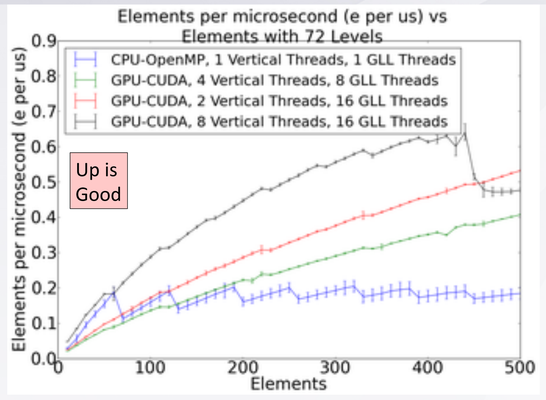


What is Kokkos?

- Kokkos is a C++11 Library for performance-portable node parallelism
- Kokkos users achieve portability by programming to Kokkos abstractions (e.g., parallel algorithm, data layout, hierarchical memory location, and compute resource)
- Kokkos library provides back ends for efficient execution of kernels on all current and upcoming DOE HPC architectures
- Application implements parallelizable *kernels* using the Kokkos abstractions
  - The application is responsible for writing thread-scalable, high-performance kernels
- Supported architectures: Multicore CPU, Intel Xeon Phi, NVIDIA GPU, …

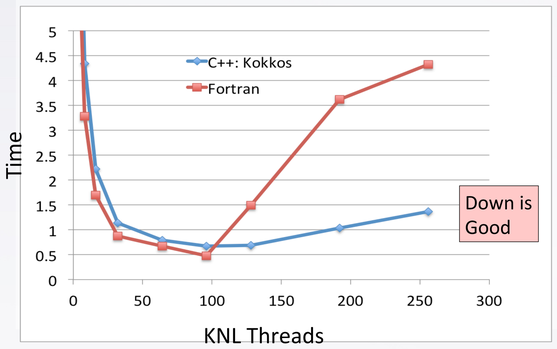


Above: optimization of mini app over time

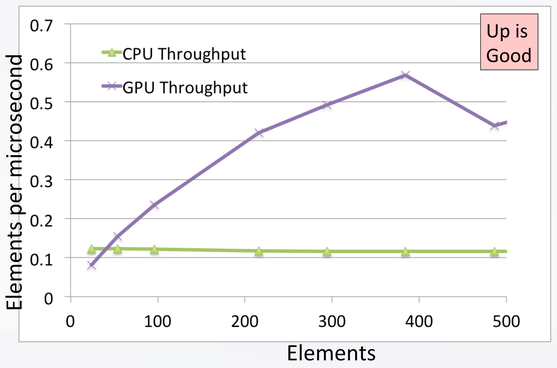


First, above, in mini app GPU and CPU Performance of dynamics

Below, KNL performance in HOMME (preqx)



Second, below, in HOMME (preqx)



HOMME Mini App

HOMME Mini App:

- Navier-Stokes Kernels from Hydrostatic HOMME
  - No MPI – mini app is just for on-node parallel performance
  - No Hyperviscosity, Transport, or Remap
  - Chose flat array layout (mimicking OpenACC tracer code)
- Wrote multiple versions to compare to extracted Fortran:
  - C++; C++ w/ static arrays; Fortran w/ fused loops
  - Kokkos with multiple kernels; Kokkos w/ fused loops

HOMME Schematic:

Mini App

- Timestep Loop
  - Dynamics Timestep Loop
    - 5 Runge-Kutta stages
      - Dynamics time step (Navier-Stokes)
      - MPI communication
    - Hyperviscosity Loop
      - Bi-Harmonic Loop
        - Laplace Operator
        - MPI communication
  - Tracers Timestep Loop
    - 3 Runge-Kutta stages
      - Transport (Hyperviscosity; Advection; Limiter)
      - MPI communications
  - Vertical Remap

Serial Runs of mini app on CPU: Relative timings compared to Fortran
100 Elements; 72 Vertical levels, 16 GLL points per element

Fortran	C++ : Static Arrays	C++ : Pointers	Kokkos, first attempt	Kokkos, fast GPU version
1.0	1.007	1.13	1.71	2.33

Small is Good

Summary and Conclusions

Summary:

- *This is a work in progress*: new team, new code, ~9 months
- Mini App of HOMME Kernels greatly increased productivity
  - Rapid exploration of data structures
  - GPU performance greatly improved
- Single code base works on GPU, CPU, KNL
  - GPU capability for dynamics is new for ACME
- Kokkos-version lags Fortran on CPU, KNL (30-50%)
  - This discrepancy will be worked -- Not acceptable for Kokkos vision
  - *This is "Bug du Jour", not the take home message*

Conclusions, so far:

- Kokkos implementation allows us to run on GPU without PGI compilers on the CPUs on Titan/Summit.
- Getting good GPU performance takes considerable effort and expertise
  - Needs to be done in mini app with tests for rapid code iterations
- Execution environment as important as code for performance
- Working with ASCR/ASC/ECP tools greatly leverages our investment and has path to talented CS experts
- Questions: What results are needed before…
  - …we swap HOMMEXX for HOMME?
  - …we refactor other components?

ACME Accelerated Climate Modeling for Energy

For additional information, contact:
Andy Salinger
CMDV Software Project

Sandia National Laboratories
Center for Computing Research
(505) 845-3523
agsalin@sandia.gov

climatemodeling.science.energy.gov/acme

U.S. DEPARTMENT OF ENERGY