

Titan Summit

8/17/2011



**Center for Accelerated
Application Research
(CAAR)**

| Application Readiness Team | |
|----------------------------|--------|
| Rick Archibald | ORNL |
| Jeff Larkin | Cray |
| Ilene Carpenter | NREL |
| Paulius Micikevicius | NVIDIA |
| Matthew Norman | ORNL |
| Valentine Anantharaj | ORNL |

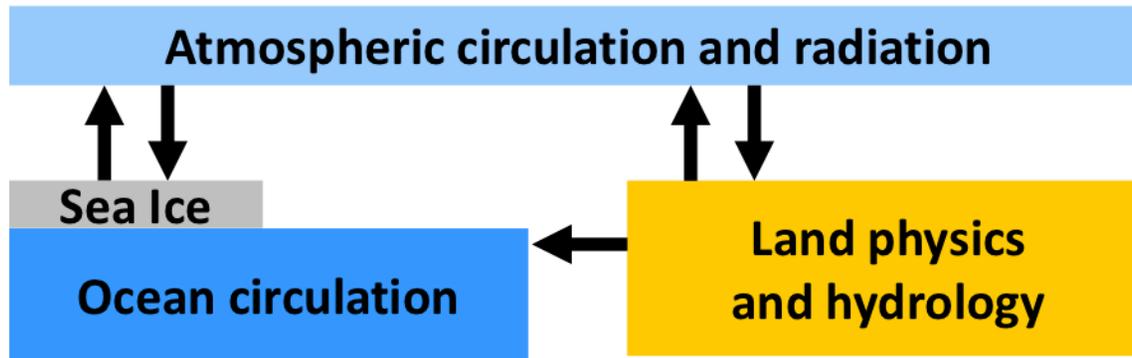
**PROGRESS TOWARDS ACCELERATING
CAM-SE ON HYBRID
MULTI-CORE SYSTEMS**

Presented by: Rick Archibald

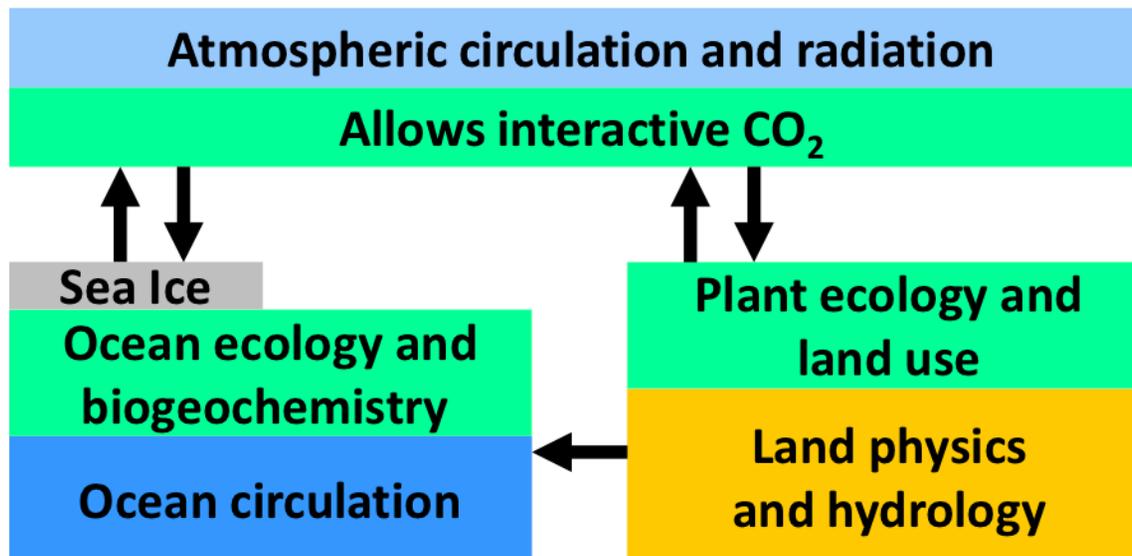
Earth System Model Description

'Climate and Ecosystems', John Dunne, 16th Annual CESM Workshop, 2011.

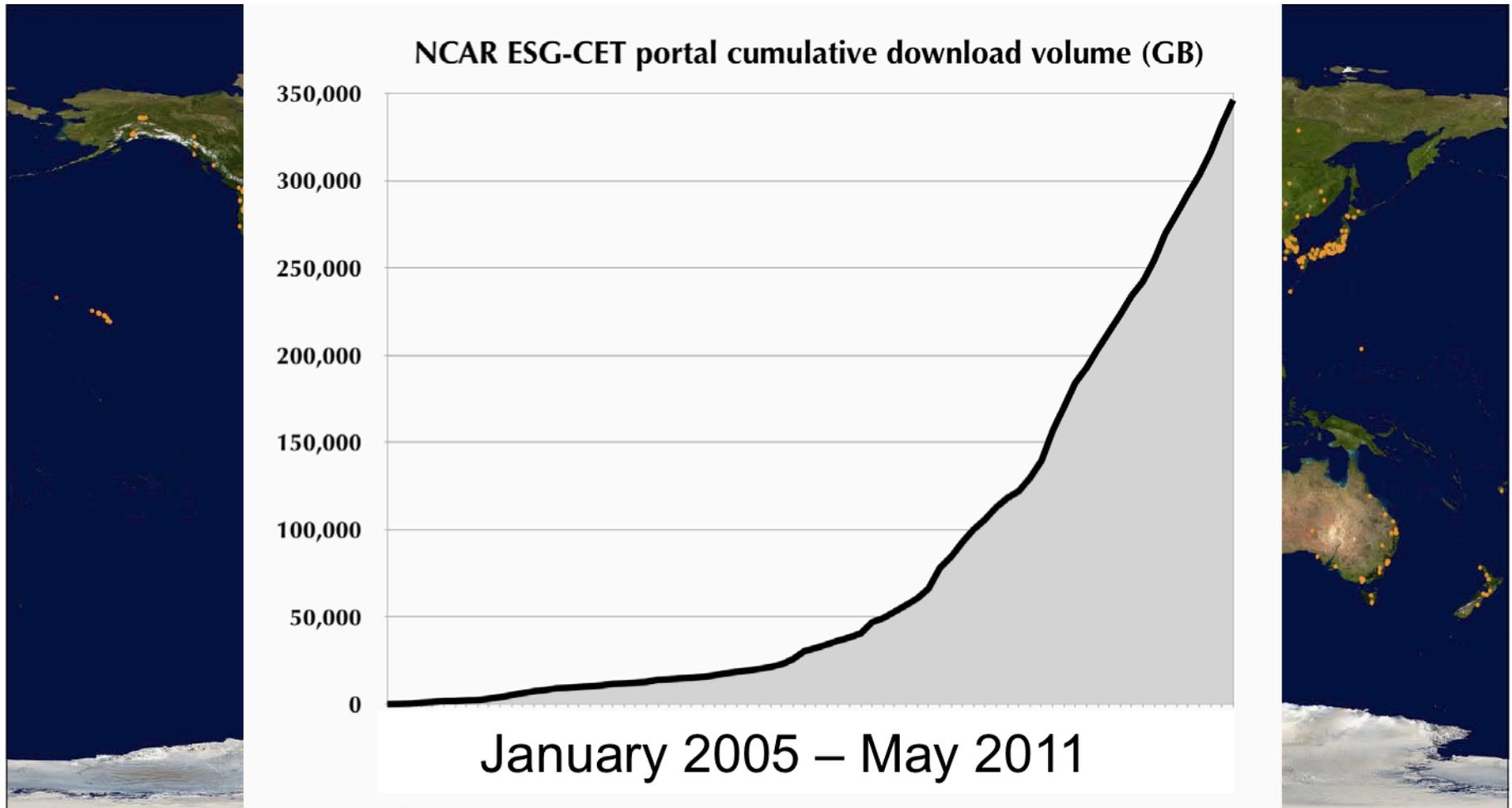
A Climate Model closes the radiative and hydrologic cycles



An Earth System Model closes additional cycles as well



A Community Resource



Over 3,000 sites from 130+ countries
>320 TB since January 2008
>1500 Registered Users of CESM1.0

Courtesy Gary Strand



20 June 2011

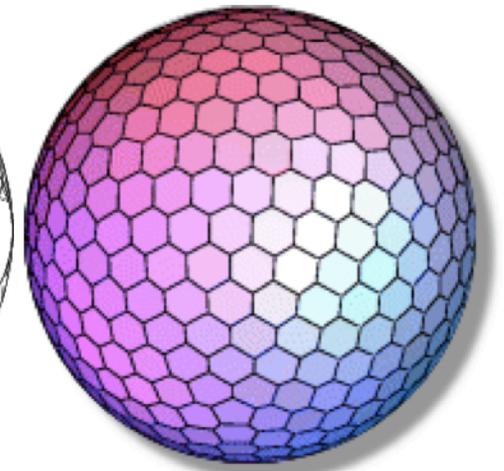
The Community Earth System Model
16th Annual Workshop

Jim Hurrell
jhurrell@ucar.edu



New Capabilities

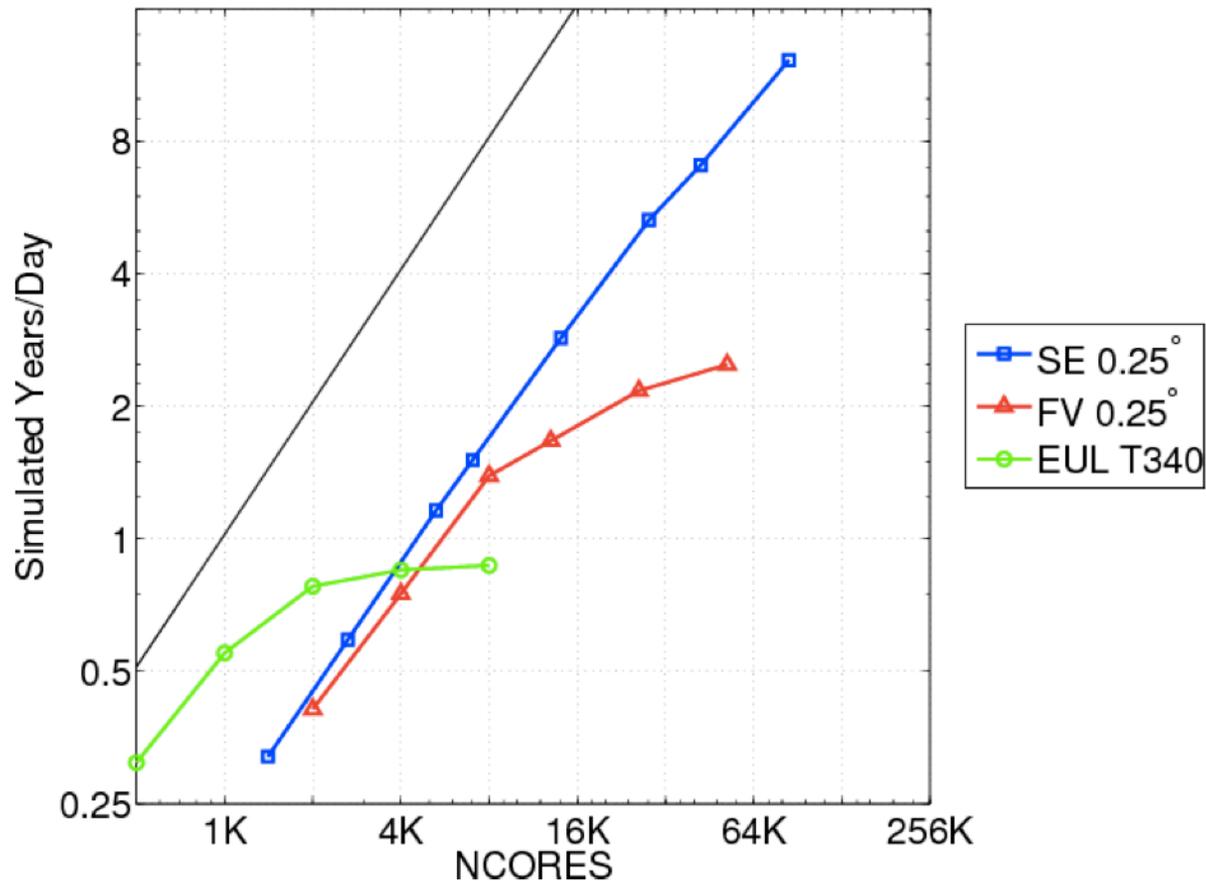
- No Polar Filter
- Advection
- Energy Conservation
- Non-hydrostatic
- Improved Scalability
- Local Mesh Refinement



‘New capabilities and new dynamical cores in CAM’. Mark Taylor, 16th Annual CESM Workshop, 2011.

Scalability: CAM4 1/4 degree (3 tracers)

CESM1 F1850, ATM component, BGP



- Faster performance at high resolution on parallel computers
- **Not cheaper** - cost in terms of core-hours is the same.
- CAM-Eul the most efficient (cheapest), but with lowest peak SYPD.

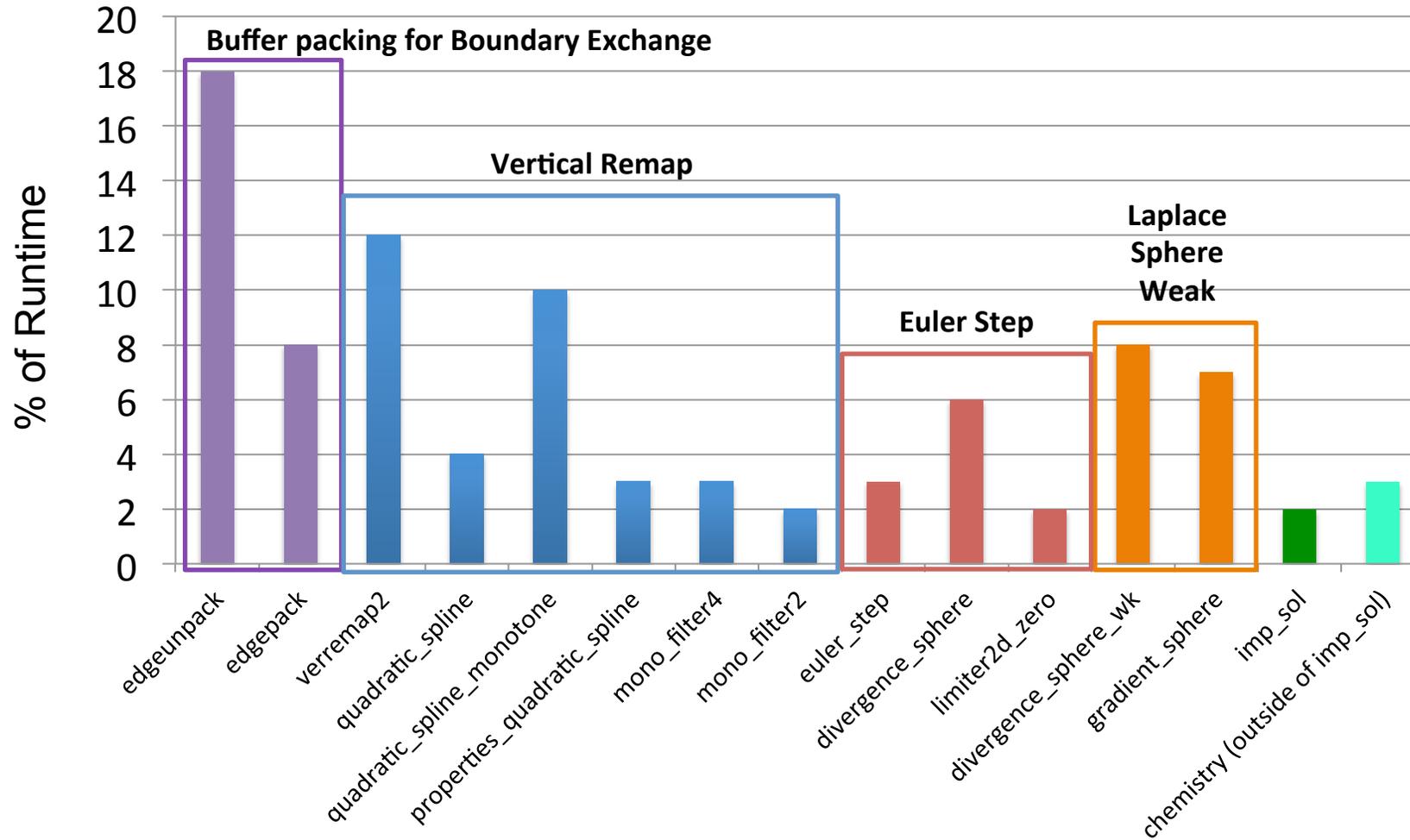
Target Problem

- 1/8 degree CAM, using CAM-SE dynamical core and Mozart tropospheric chemistry. Land model will be run on CPUs.
- Why is acceleration needed to “do” the problem?
 - When including all the tracers associated with Mozart atmospheric chemistry, the simulation is too expensive to run at high resolution on today’s systems.
- What unrealized parallelism needs to be exposed?
 - In many parts of the dynamics, parallelism needs to include levels and chemical constituents.

Profile Generation

- The science problem of interest could not be run at the time the project started because Mozart chemistry depended on using a latitude/longitude grid and CAM-SE uses a cubed-sphere grid.
- A set of jobs that spanned the aspects of this problem were run, which allowed us to make a **performance model** for the target problem.
- Recently this limitation has been overcome and we can now run the problem of interest. The new profile confirms that our performance model directed us to work on the right part of the code (tracer advection).

Projected Profile of Runtime



Kernels extracted

- Vertical remapping
- Euler_step + limiter2d_zero
- Laplace_sphere_wk
- Small kernels from dynamics (gradient, divergence, vorticity etc.), used by euler_step and laplace_sphere_wk
- Tendency physics before coupling (for compiler analysis, not run-able).
- Chemistry implicit solver (smaller than expected % of final runtime, further development put on hold)

Method of acceleration

- CUDA
 - We primarily used PGI CUDA Fortran for HOMME kernels.
- Directives
 - Recent studies with the Cray compiler on the HOMME kernels show that the compiler directive approach can provide as good or better performance as manual tuning with CUDA Fortran.
- Libraries
 - No significant math library use

Method of acceleration: Why?

- **CAM-SE** dynamical core has the same loop and index patterns appear repeatedly, which makes CUDA work tractable.
- **Mozart chemistry** code developers are likely to accommodate code changes. Both CUDA and directives will be explored. Implicit solver code is generated by a pre-processor, so GPU-specific code can be generated without affecting CPU version.
- **Physics** code will need to be accelerated using directives if this is to be run on GPUs
 - to prevent version bifurcation, involves many routines written by many different people who have no knowledge of accelerators
 - profile for the physics is extremely flat
 - physics is less than 1% of target job so low priority now
 - Directives are the only method that potentially allow one to use GPUs without causing unacceptable changes in the CPU base code.

Plan of work – what has been done?

- Examination of code structure and data types.
- Created test cases for profiling and performance model development.
- Created macro-kernels for some of the most time consuming parts of the code to enable people unfamiliar with CESM to assist with project.
- Created low level kernels from CAM-SE, then moved up the call tree to higher level kernels.
- Optimized vertical remapping, which improved performance on CPU and enabled use of the GPU.
- Implemented asynchronous data transfers with standalone kernels. Determined that overlapping data transfers with computation or leaving data on GPU is critical.
- Explored doing packing/unpacking for boundary exchange on GPU. This is sub-optimal.
- Separated on-GPU boundary exchange (interior edges) from off-GPU boundary exchange, to minimize data transfers back to CPU during tracer advection.

CPU code:

```
Loop over elements
  Loop over advected constituents (q)
    Loop over j
      Loop over i
        Loops over levels
          all computations from quadratic_spline and
          quadratic_spline_monotone manually inlined
```

GPU code:

```
Transfer element-independent data to GPU
Create chunks of 6 elements
```

```
Decompose work into thread blocks:
```

```
tblock=dim3(nv,nv,6) - columns associated with 6 elements
tgrid=dim3(qsize,1,1) - each block has a different value of q
```

```
Loop over chunks:
```

```
Copy data from element structure to local arrays
Transfer data to GPU
Call remap_Q_kernel<<<tgrid,tblock>>>(…)
Transfer results back to CPU
Copy data back to elem structure
```

```
attributes(global) subroutine &
  remap_Q_kernel(Qdp,hyai,hybi,ps_v,dpdn,ps0,dt)
```

```
  i = threadIdx%x
  j = threadIdx%y
  ie = threadIdx%z
  q = blockIdx%x
```

```
! each thread has one column of points for one value of q
```

```
  Loops over levels
```

```
    all computations from quadratic_splime and
    quadratic_spline_monotone manually inlined
```

```
end subroutine remap_Q_kernel
```

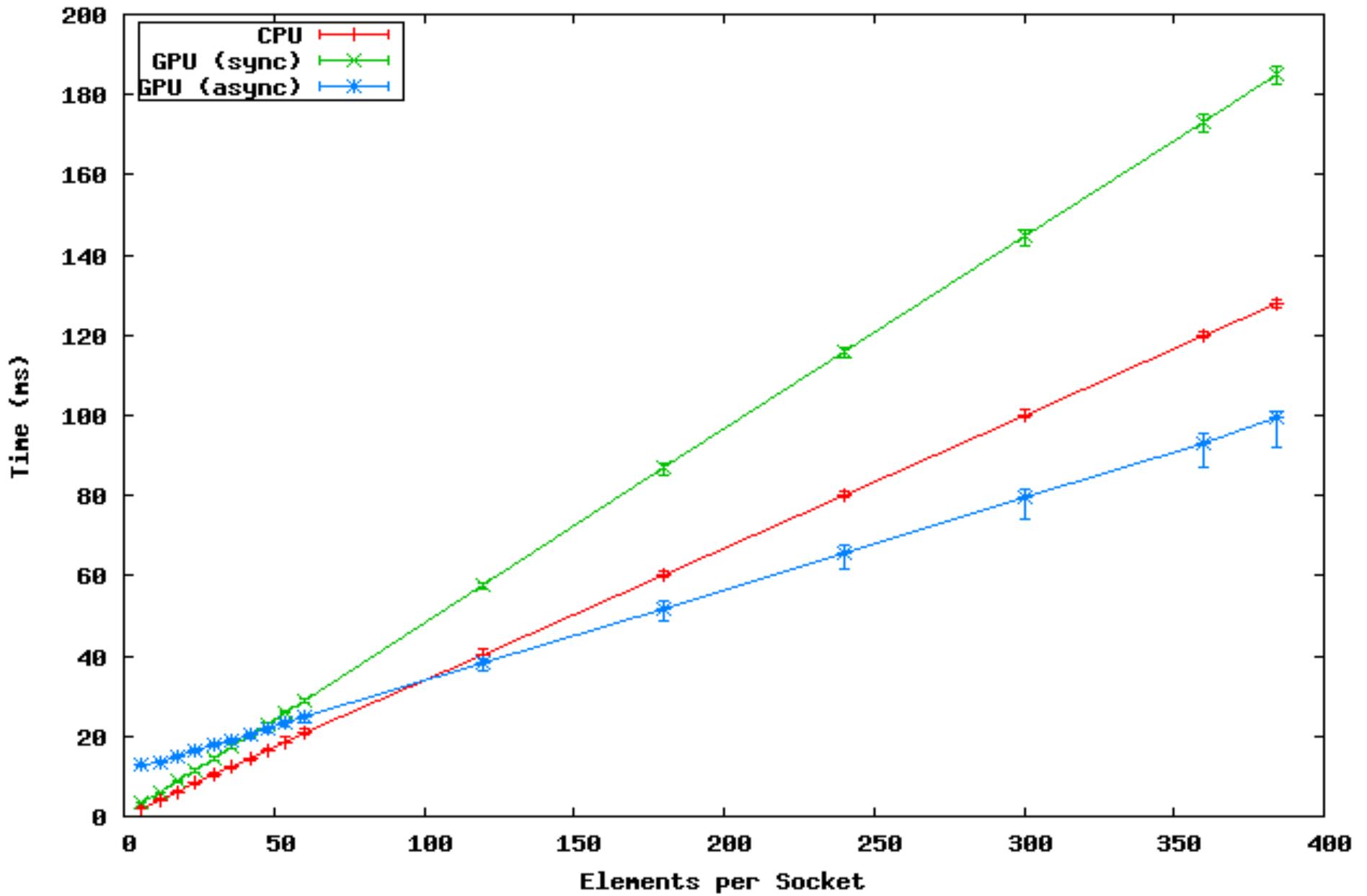
Vertical Remapping

- The mass and momentum variables in CAM-SE are conservatively remapped at the end of each time step following Zerroukat 2005 and 2006.
- Computation for each column is independent of other columns, which immediately offers parallelism for $nv \times nv \times qsize \times nelem$ threads
- Each kernel launch has $qsize$ thread blocks, and each thread block is $nv \times nv \times nelem$ threads
- Optimal memory access patterns – each group of $nv \times nv$ threads accesses a contiguous memory region since level data is contiguous for each element and tracer

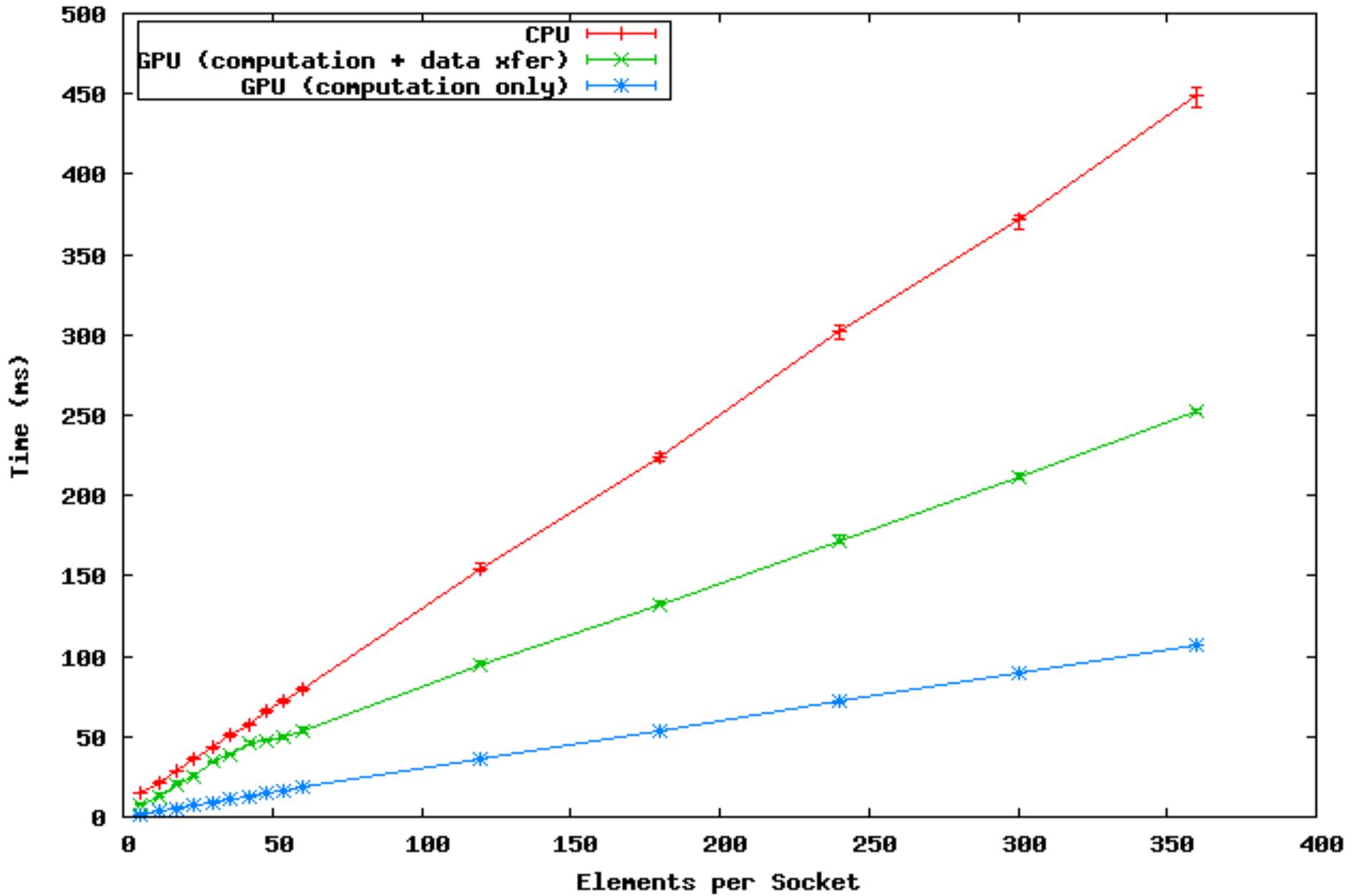
Plan of work – what remains to be done?

- Continue to refine profiles as inefficient code is rewritten and work on routines that take a significant percentage of the runtime.
- Optimize on-GPU boundary exchange (pack and unpack).
- Optimize buffers containing boundary data that need to go to CPU to generate MPI messages.
- Move up a level with GPU kernels to include data that can be left on GPU using new boundary exchange methods.
- Integrate GPU kernels into HOMME trunk (and via that, into CESM)

Euler Step Comparisons



Vertical Renap Comparisons



Summary

- The community is moving towards high-resolution modeling with atmospheric chemistry.
- We have demonstrated that such a run is dominated by tracer advection.
- We have demonstrated that enough parallelism exists to run tracer advection on the GPU.

Additional help from

Jim Rosinski, Mark Taylor, John Dennis, Kate Evans, Oscar Hernandez, Jim Schwarzmeier, Tom Court, Abdulla Bataineh

