Improving CAM Throughput at Scale

Art Mirin

Lawrence Livermore Nat'l. Lab.

Pat Worley

Oak Ridge Nat'l. Lab.

June 15, 2009

Context of this research

- Polar singularity limits ability to effectively domain-decompose in longitude
- Long-term solution is to use more favorable grid
  - cubed sphere (finite-volume, spectral element)
- In near term (including IPCC AR5), we need to live with the longitude-latitude grid
- Approach: add parallelism and address scaling bottlenecks

CAM Throughput has Improved

We have more than doubled the performance of the Community Atmosphere Model on the Cray XT4/5 and are seeing similar improvements on the IBM BG/P. This has come about through a combination of adding additional parallelism, enabling different sections of CAM to execute at their own process count, implementing improved communication protocols particularly relevant at scale, and removing other scalability bottlenecks.

***Throughput improvement is problem-dependent.**
*** Work carried out over past 2.5 years under SciDAC-2.**

Improvements reported at AMWG 2008

- **We added additional parallelism and enabled different sections of code to execute at their own process count**
  - **allow one vertical level per subdomain**
  - **assign more (computational) processes to physics than dynamics**
  - **advect multiple tracers concurrently**
  - **larger longitude-latitude than latitude-vertical decomposition**
  - **overlap of main dynamics and tracer advection subcycles**

Resulting scalability bottlenecks

- Communication inefficiencies due to large number of messages
- Less ability to hide communication latency
- Inefficiencies computing global sums
- Input/output inefficiencies
- Memory overflows (in particular associated with communication)

We have removed communication bottlenecks at scale

- **Improvements to FV dynamics transposes (*mod_comm*)**
  - **all-to-all option**
  - **hypercube-based (*swap*) ordering of communications**
  - **ability to transpose 2 variables simultaneously**
  - **ability to transpose arbitrary number of tracers concurrently**
  - **handshaking (wait to issue send until matching receive is issued)**
  - **throttling (limit number of outstanding requests)**
  - **blocking vs. non-blocking send**
- **Improvements go hand-in-hand with those in dynamics-physics transposes and spectral dycore communications**
- **Different code sections can use different options**

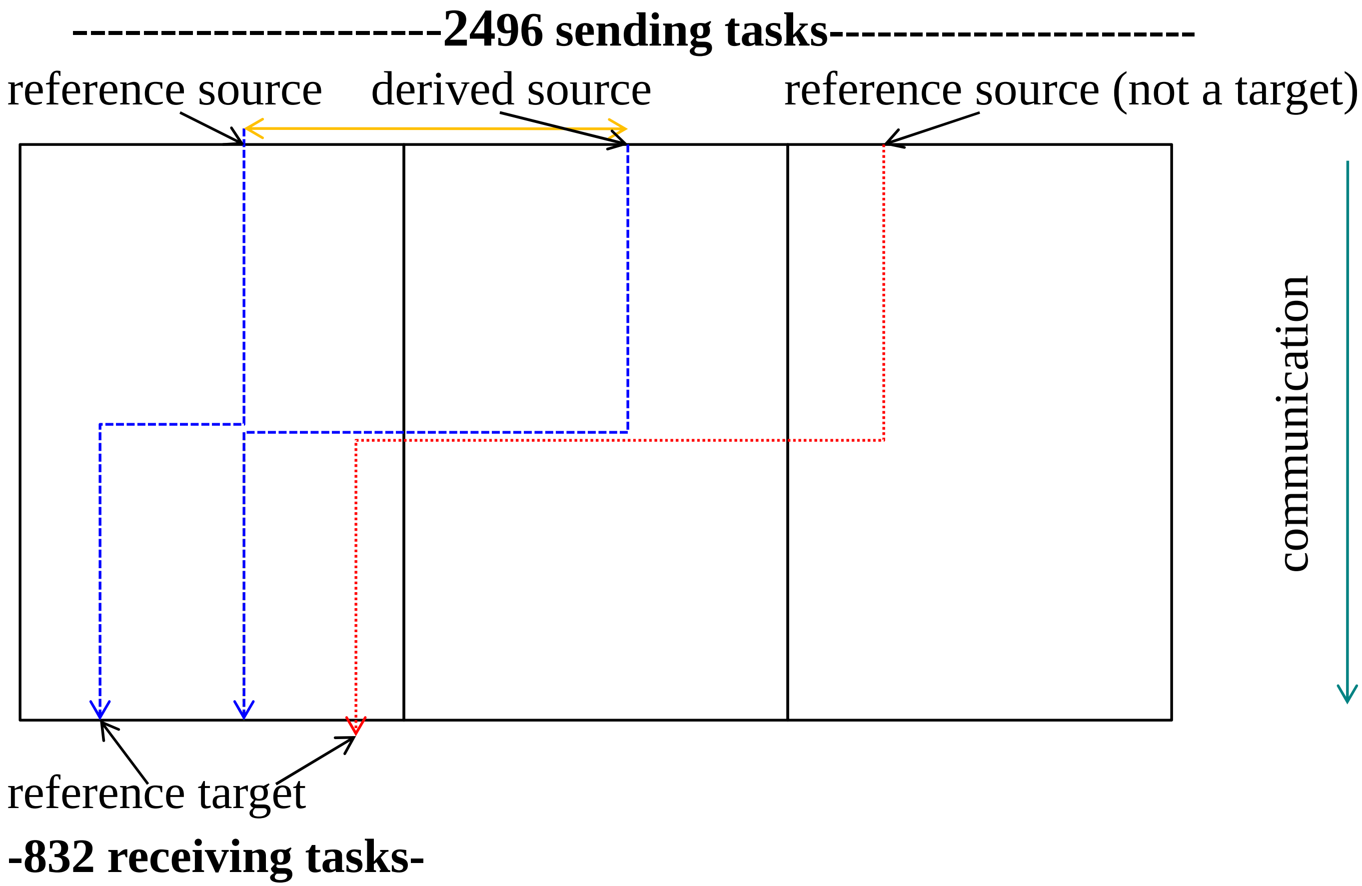
Communication bottlenecks at scale, cont.

- Apply flow control (handshaking, throttling) to global gathers
- In PIO layer, apply flow control to box rearranger and netcdf write
- Introduction of handshaking could require throttling
  - handshaking alleviates MPI buffer storage by controlling incoming messages
  - handshaking introduces additional messages, putting pressure on the available MPI storage for message accounting
  - throttling, by limiting the number of simultaneously active messages, alleviates pressure on MPI storage
- The handshaking messages themselves incur cost; hence, handshaking should be used only when required
  - handshaking is typically on by default only for gathers

Example communications bottleneck

- **FV 1-deg grid: transpose from 39x64 longitude-latitude decomposition to 64x13 latitude-vertical decomposition**
  - **one-third of target tasks show order-of-magnitude larger elapsed time (overall code runs twice as slow)**
  - **reproducible on Atlas, Jaguar and BG/P**
  - **the tasks that take longer are precisely the ones that are posting receive requests to source tasks who themselves are also target tasks (all tasks are source tasks; only 1/3 of tasks are target tasks)**
  - **solution is *handshaking*: do not initiate send until target has communicated that it is ready to receive; this reduces contention**
  - **problem is less likely to occur with equal-size decompositions because scenario is better balanced; handshaking is usually not required in this case**

Diagram of communications bottleneck

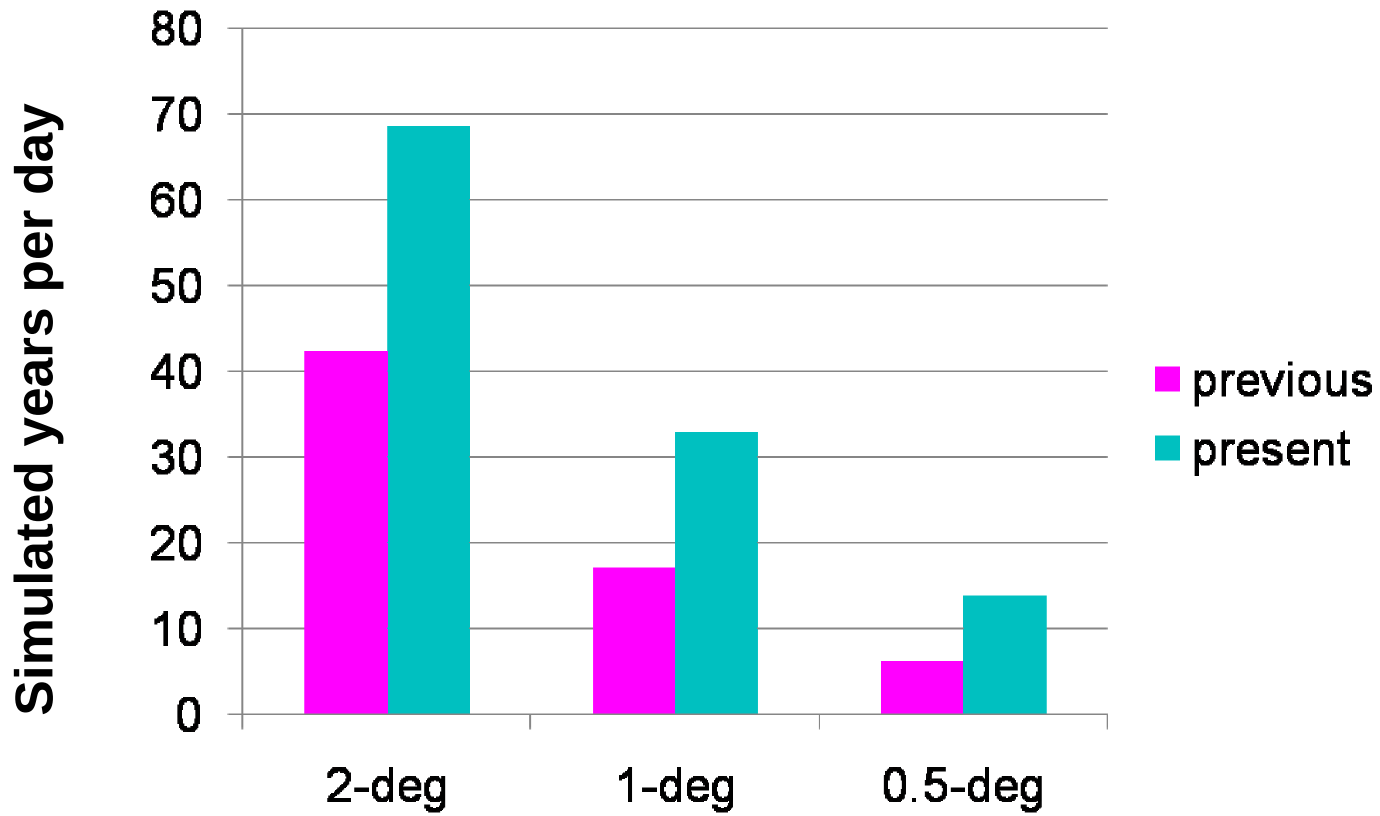


Only 1/3 of the target tasks have a source task that itself is a target; resulting contention causes delay.

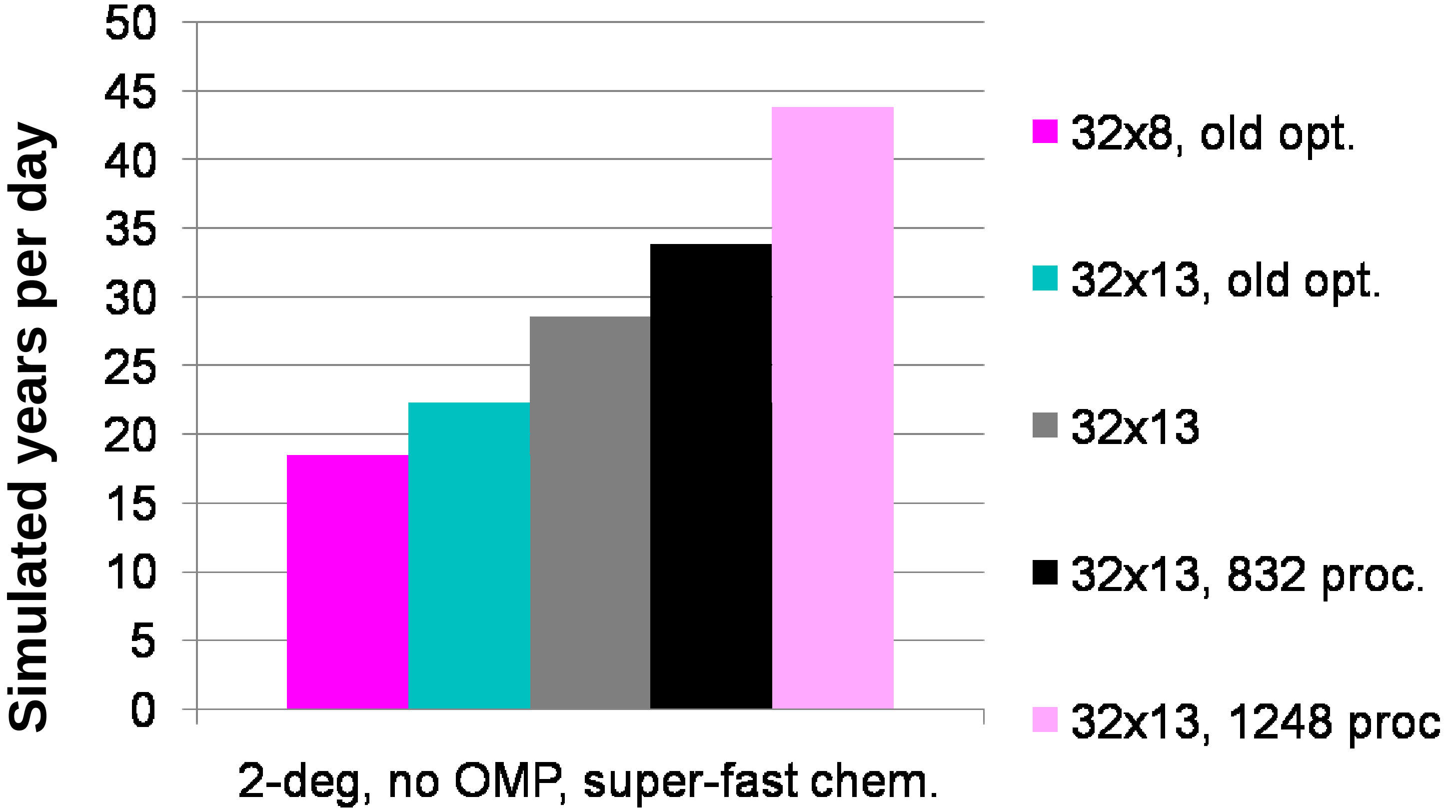
We have removed other scaling bottlenecks

- **Fast reproducible distributed sum algorithm (replaces one-process-computes algorithm)**
  - **uses 64-bit integer arithmetic to implement infinite precision algorithm**
  - **more accurate than original floating point calculation**
  - **implemented in physics and dynamics**
  - **adopted in CCSM coupler/driver**
- **Reproducible, non-transpose-based geopotential algorithm that eliminates real*16**
  - **utilizes pipeline approach**
  - **optionally replaces the transpose-based algorithm; optimal choice depends on platform, problem size and process count**
- **Elimination of memory inefficiencies**
  - **globally dimensioned arrays**

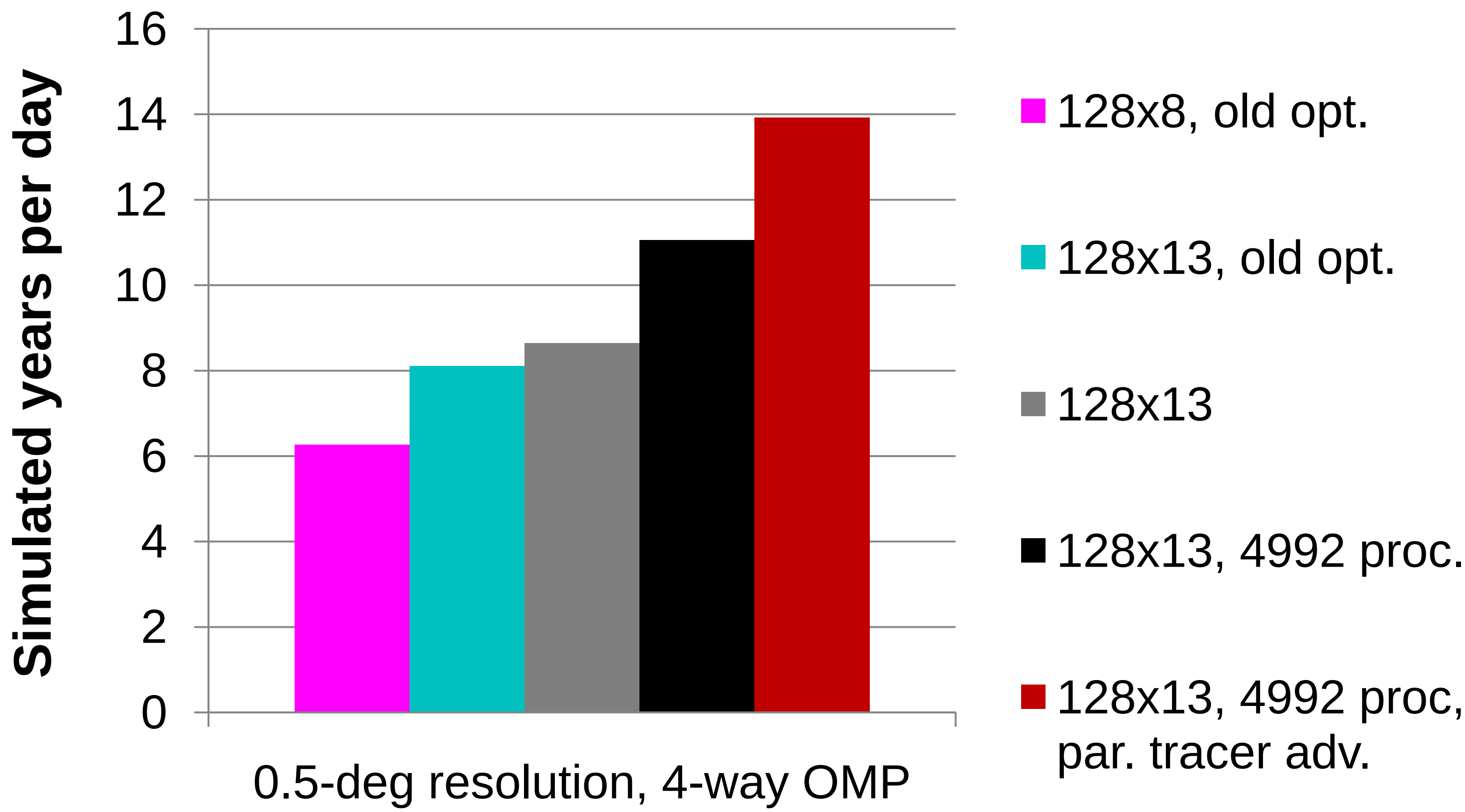
CAM with 4-way OpenMP on Cray XT4



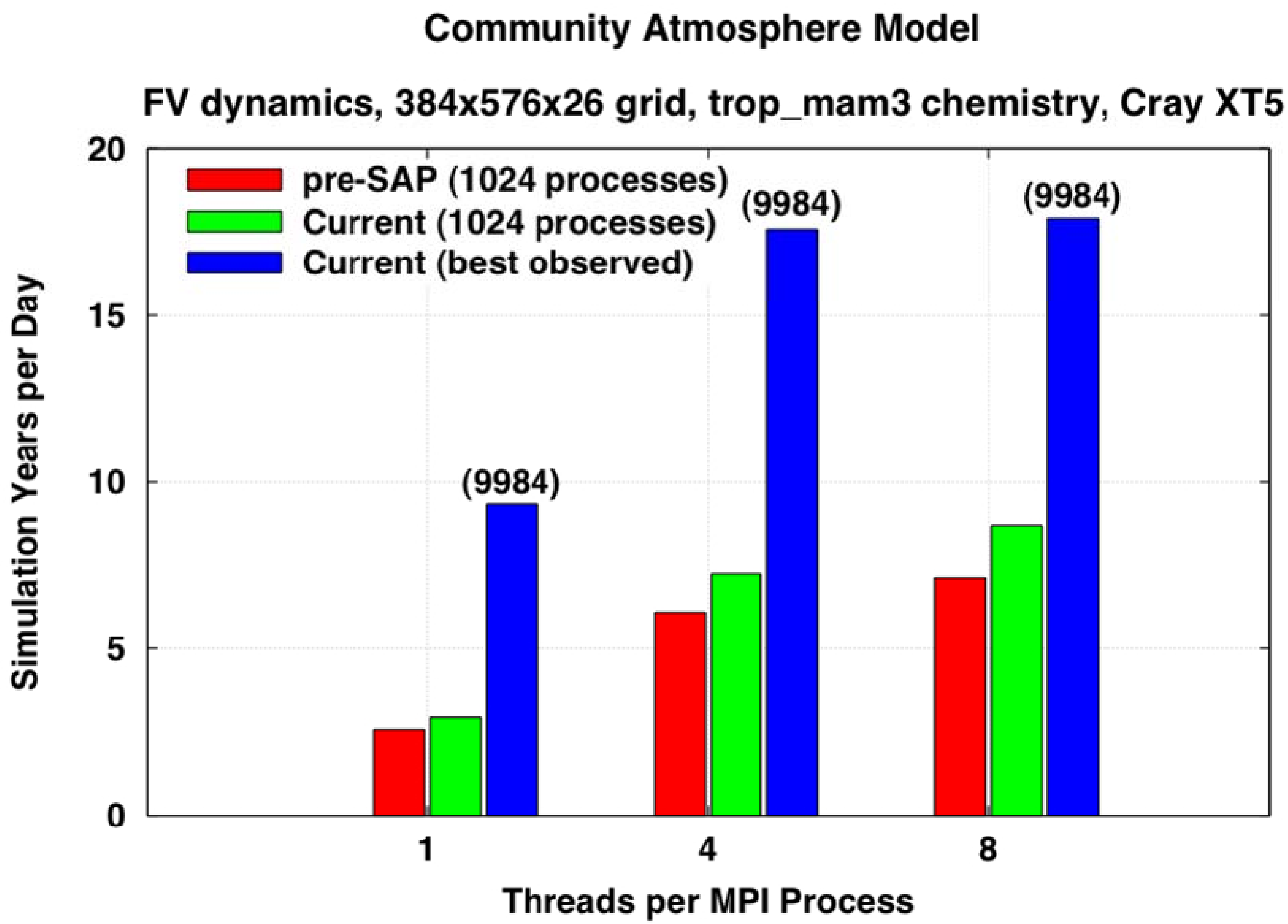
Breakdown of performance improvement



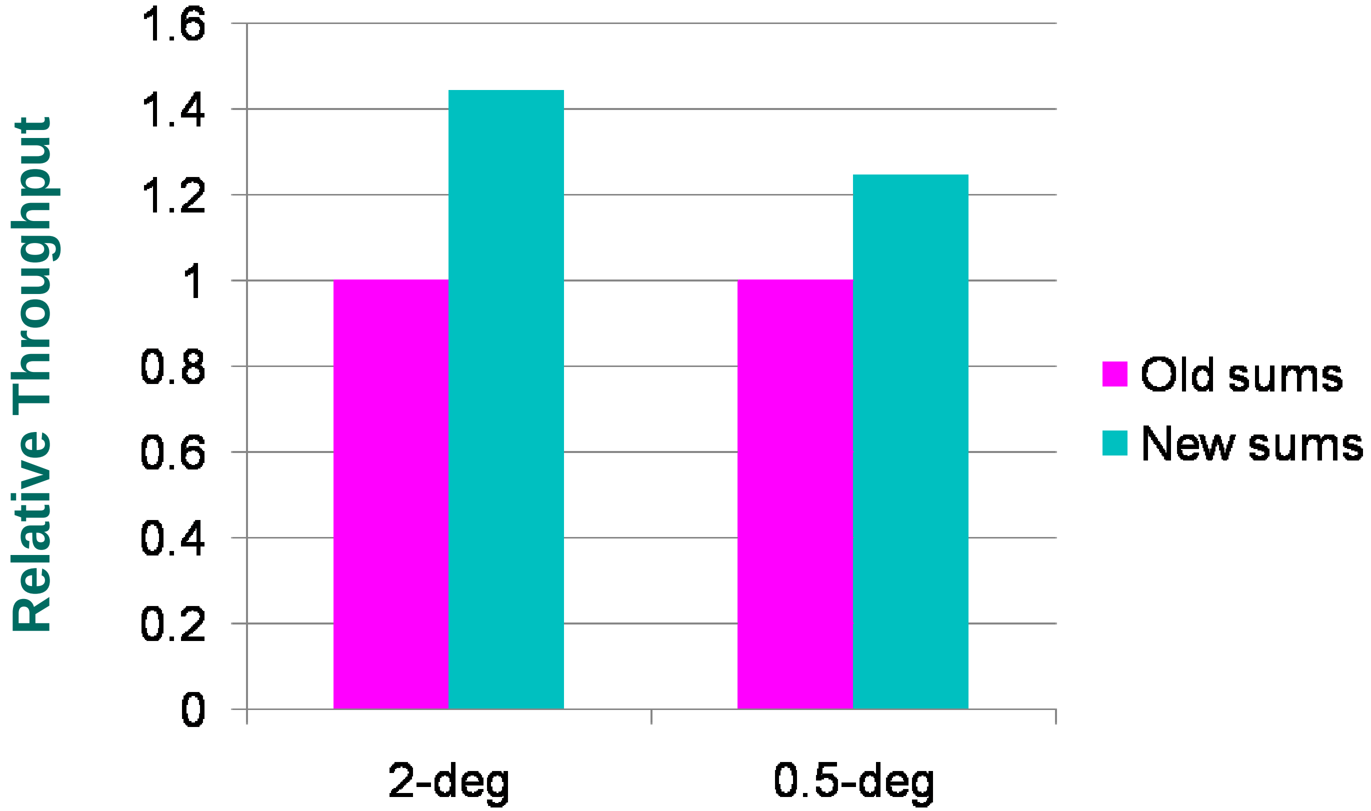
Breakdown of performance improvement



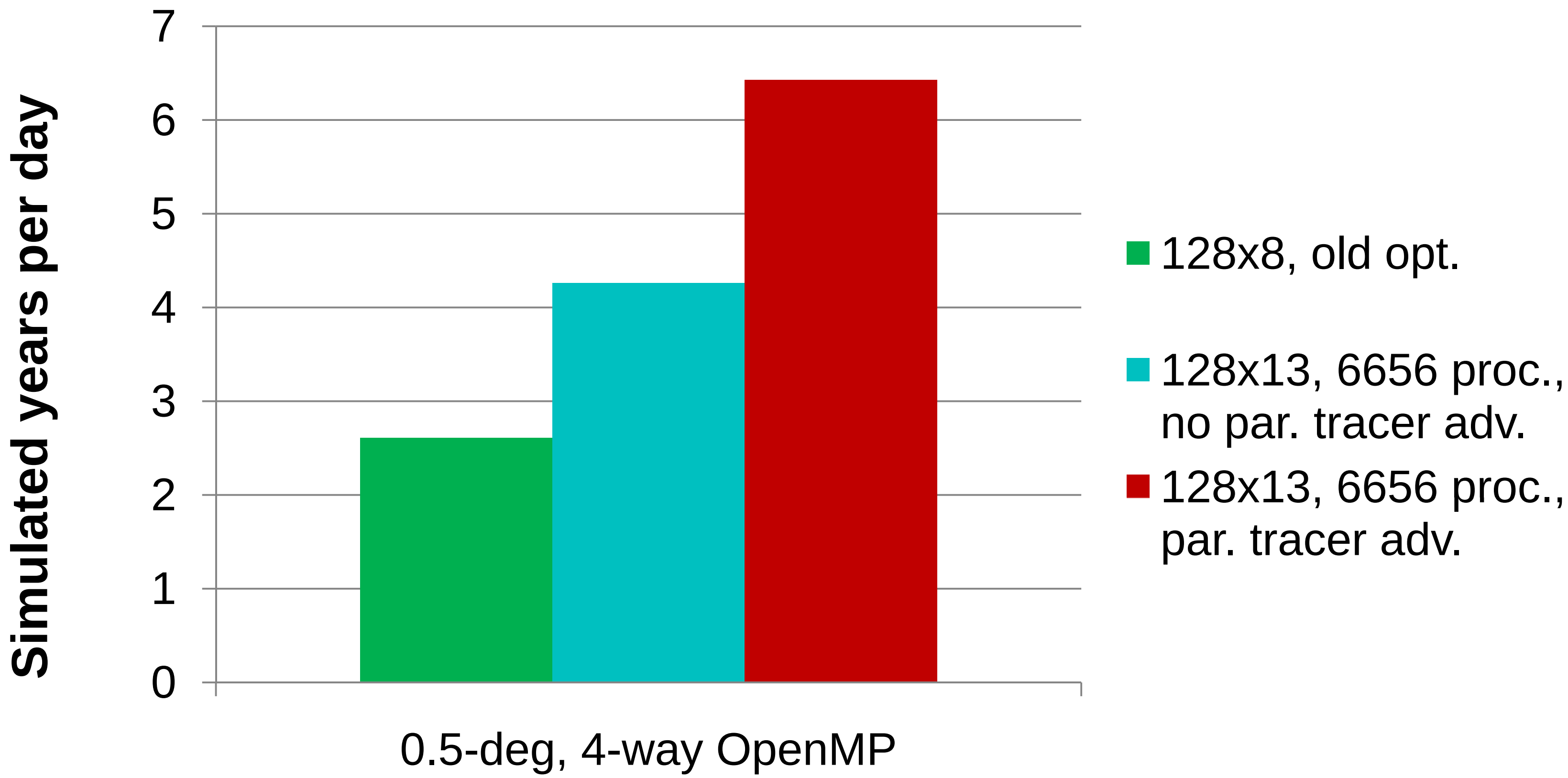
Improved throughput on Cray XT5 (vs OpenMP)



Fast reproducible distributed sums



Performance improvement with trop_mozart chemistry



High-resolution sectional aerosol simulation on BG/P

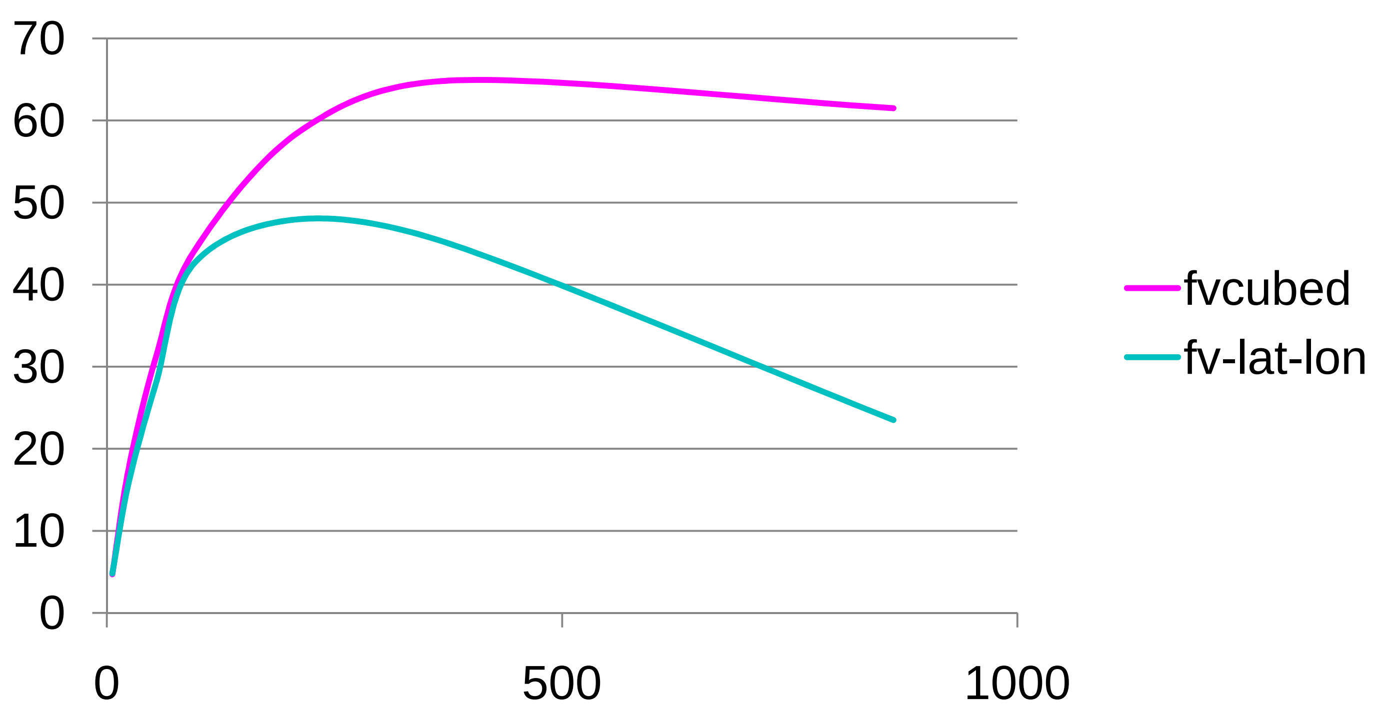
- We are hoping to carry out 0.5-deg and 0.25-deg sectional aerosol simulations on the LLNL 0.5PF Dawn BG/P
  - Dawn , which contains 36864 4-processor 4GB nodes, is the initial delivery system for the Sequoia project
- A 0.25-deg case with trop_mozart chemistry and 399 tracers executes at about 0.4 simulated years per day (excluding I/O) using 26624 processors of jaguar
  - assign 2 tasks per node since nodal Jaguar memory is double that of Dawn
  - we successfully produce history and restart dumps using pio branch of CAM
- A similar case, but using the CAM trunk with integrated PIO and having 350 tracers instead of 399, executes at 0.09 simulated years per day (excluding I/O) on 8192 nodes (32768 processors) of Dawn
- We estimate computational throughput using 32768 nodes (131072 processors) of Dawn to be about 0.18 simulated years per day

Utilization of PIO at high resolution

- PIO is required to avoid memory overflow
  - — must use new 64-bit netCDF format
- Performance is sensitive to configuration and architecture
  - — tune with respect to number of IO tasks, netcdf vs pnetcdf
- At 0.25-deg with trop_mozart chemistry and 350 tracers, using pnetcdf:
  - — time on Jaguar to write restart dump = 4 min.
  - — Dawn takes 13 min. for restart dump, 5 min. for history file
  - — the above times represent a single instantiation, hence are subject to change

Performance of FV dycores on Cray XT4

Simulated years per day vs number of tasks



Upcoming tasks in support of AR5

- **Implement and evaluate additional OpenMP in FV dycore**
  - **presently limited to number of levels per subdomain**
- **Establish default optimization settings as function of**
  - **problem type**
  - **resolution**
  - **architecture**
- **Rerun benchmark tests to understand costs of different physics and chemistry options for AR5 scenarios**
- **Determine optimal processor configurations considering**
  - **available machine cycles**
  - **maximum time to solution**
- **Benchmark, evaluate and optimize CCSM4 release on AR5 target architectures**

Other plans

- Characterize, optimize and evaluate performance at greater scale as CCSM evolves toward earth system model and targets emerging petascale systems
- Extend atmospheric model scalability improvements to other components
- Exploit enhanced parallelism (e.g., Opteron SSE, BG/P double hummer)
- Improve memory usage throughout model
- Continue support for and evaluation/optimization of dycores on cubed sphere grid
  - HOMME spectral element dycore
  - Finite-volume dycore

Acknowledgements

- Research sponsored by the Climate and Environmental Sciences Division of the Office of Biological and Environmental Research and by the Office of Advanced Scientific Computing Research, both in the Office of Science, U.S. Department of Energy, under Contract No. DE-AC05-00OR22725 with UT-Battelle, LLC and Contract No. DE-AC52-07NA27344 with Lawrence Livermore National Security, LLC.
- These slides have been authored by contractors of the U.S. Government under contracts No. DE-AC05-00OR22725 and No. DE-AC52-07NA27344, and are released as LLNL Report LLNL-PRES-413745. Accordingly, the U.S. Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes.
- This research used resources of the National Center for Computational Sciences at Oak Ridge National Laboratory, which is supported by the Office of Science of the Department of Energy under Contract DE-AC05-00OR22725.