Improvements in CAM Throughput at Scale

Art Mirin

Lawrence Livermore Nat'l. Lab.

Pat Worley

Oak Ridge Nat'l. Lab.

March 2, 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)

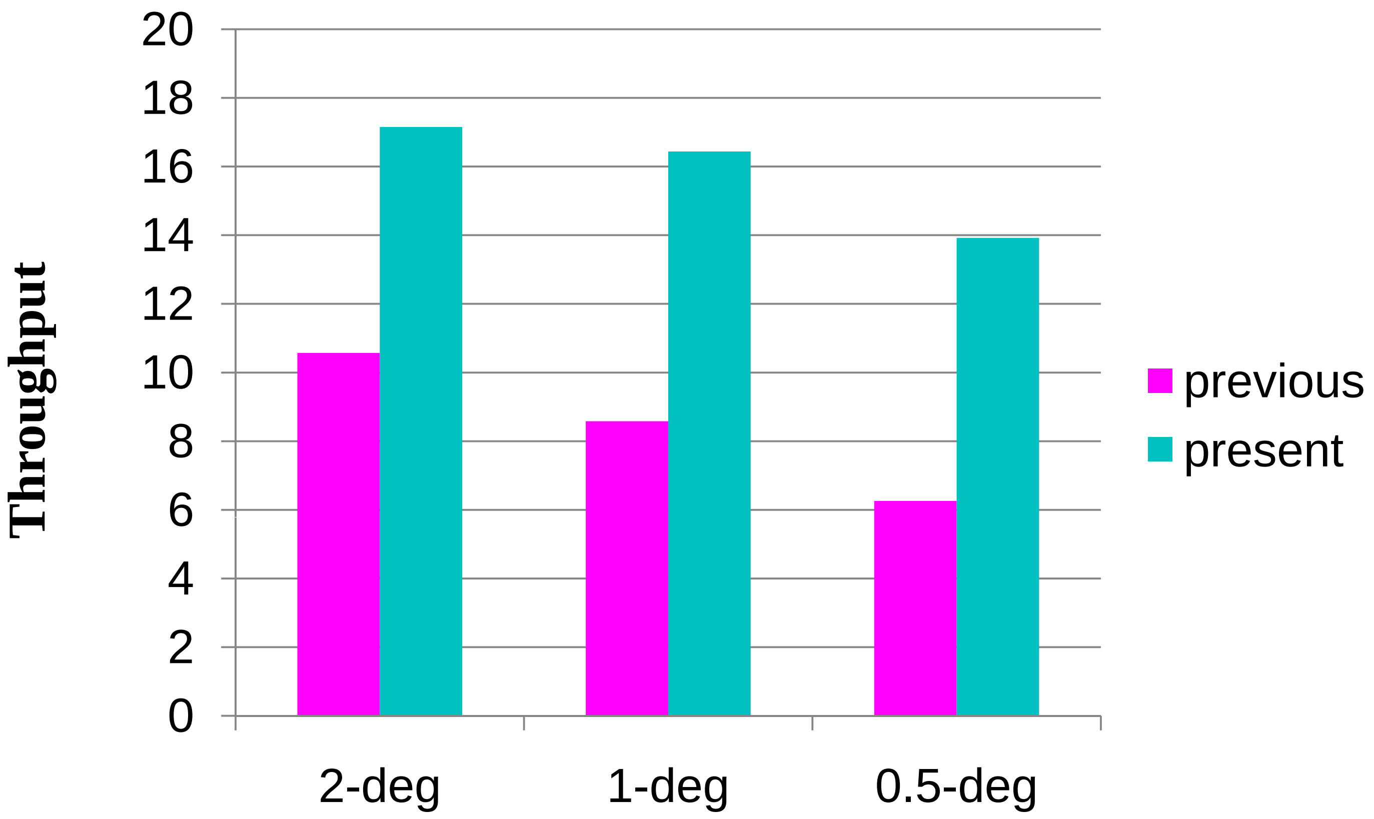
We have removed communication bottlenecks at scale

- 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
- Apply flow control (handshaking, throttling) to global gathers
- Different code sections can use different options

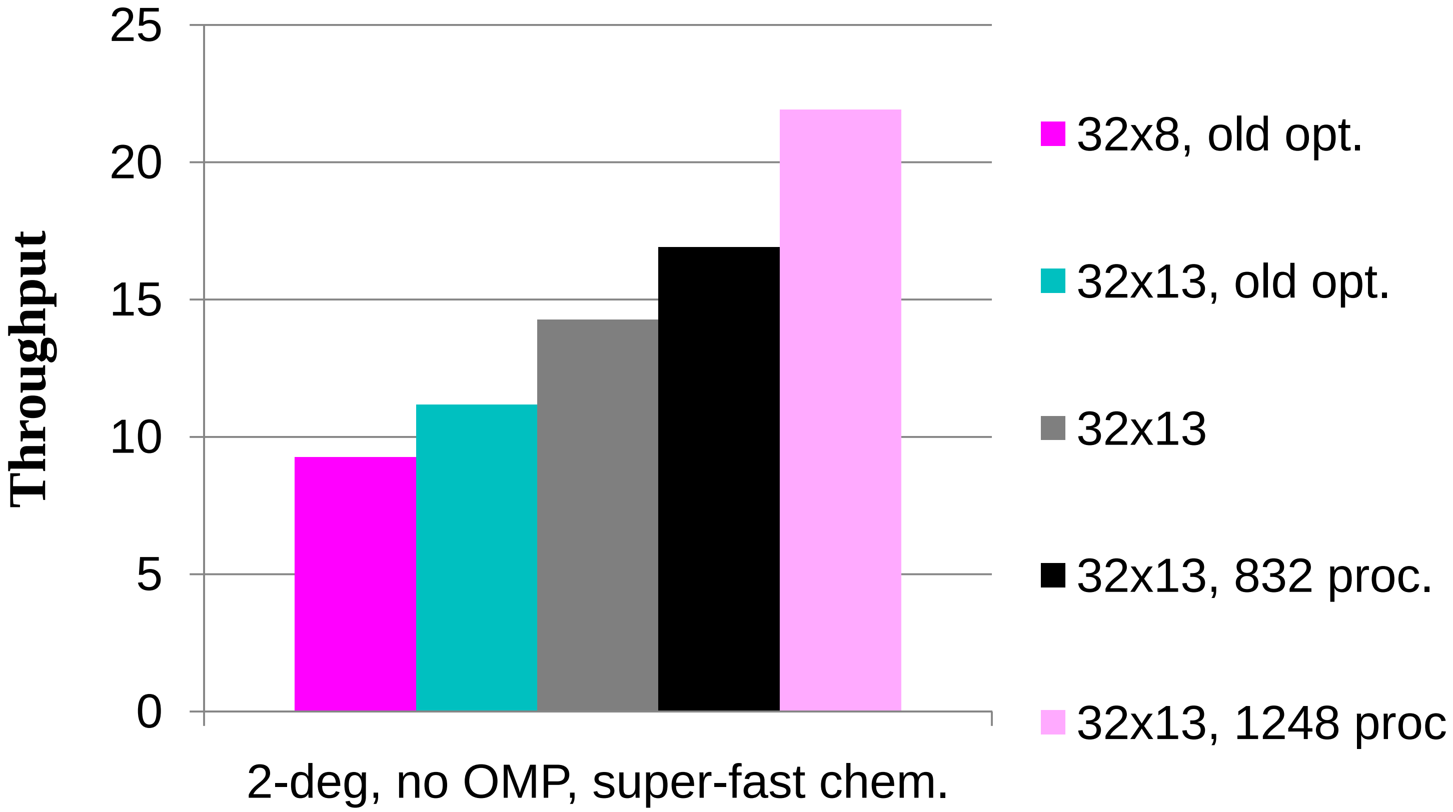
We have removed other scaling bottlenecks

- Fast reproducible distributed sum algorithm (replaces one-process-computes algorithm)
  - —used in physics and dynamics
- Non-transpose-based geopotential algorithm that eliminates real*16
- (Parallel I/O is being addressed as a wider collaboration.)

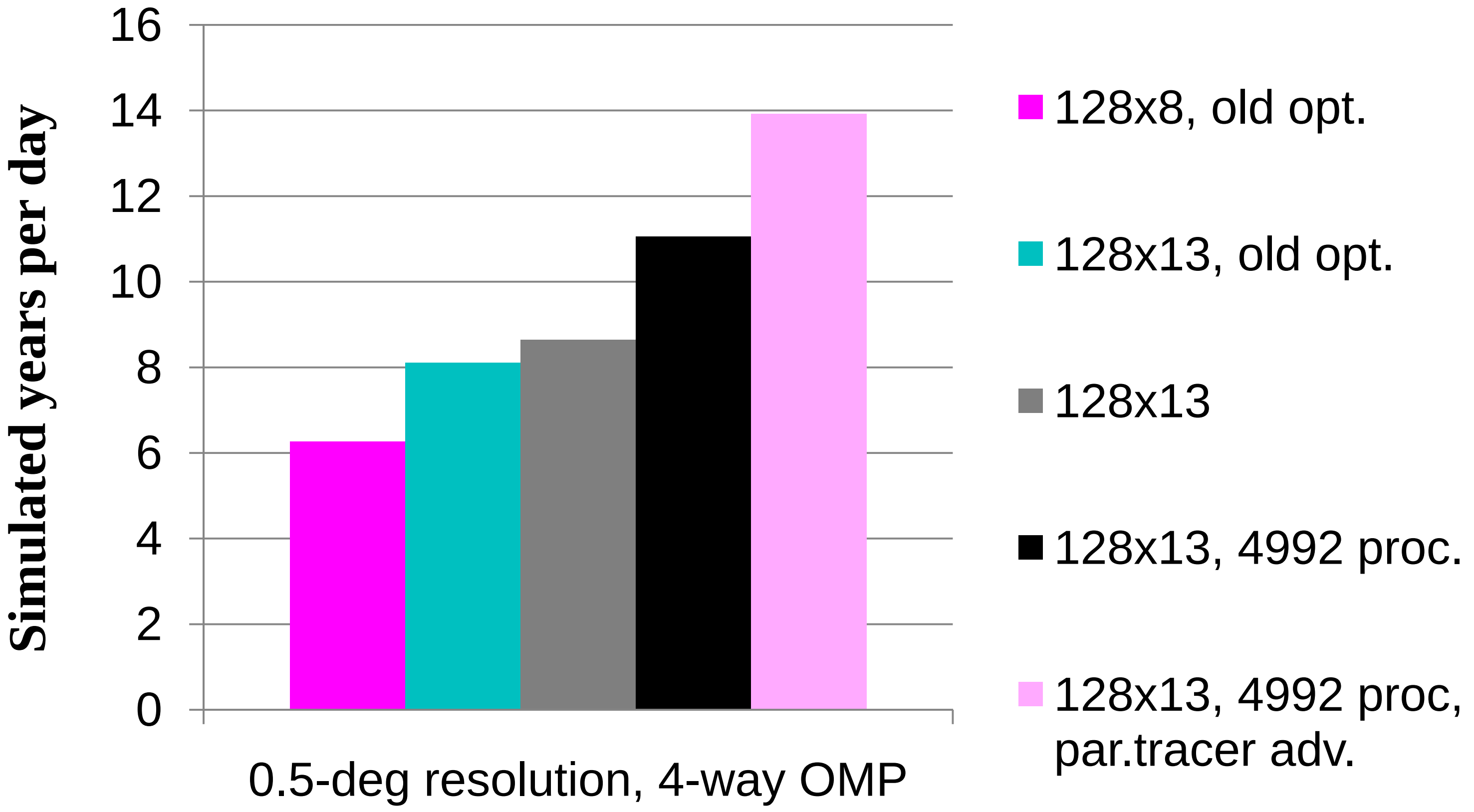
CAM with 4-way OpenMP on Cray XT4



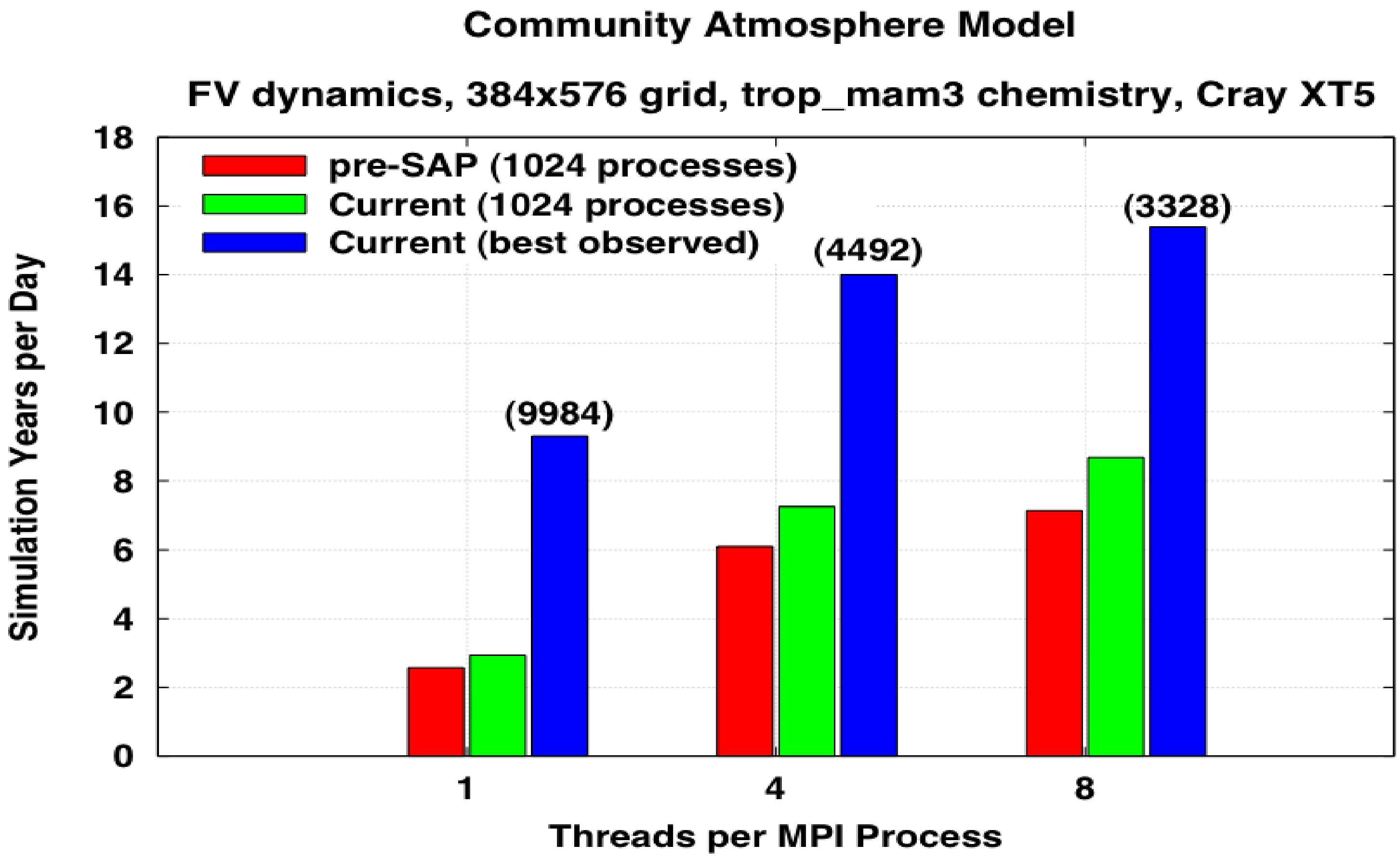
Breakdown of performance improvement



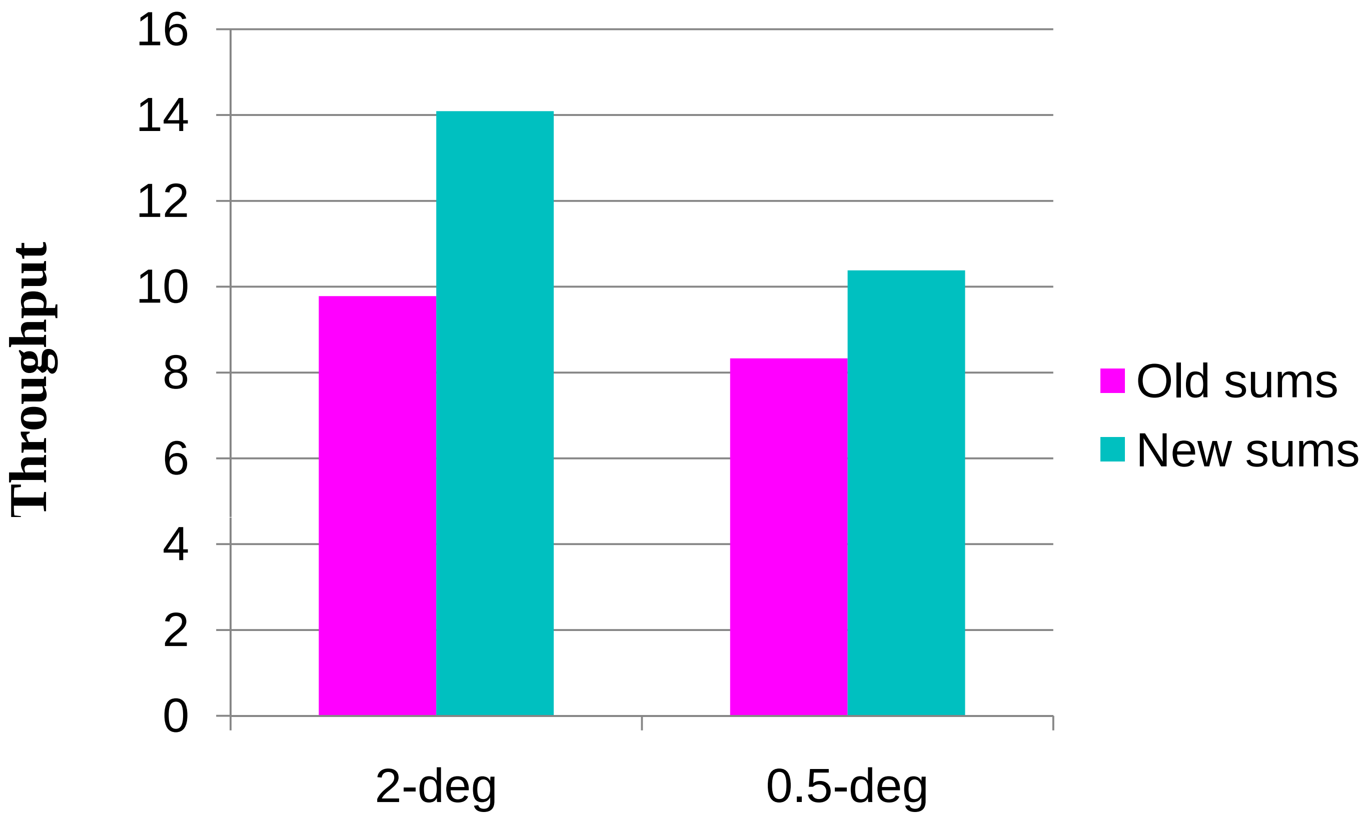
Breakdown of performance improvement



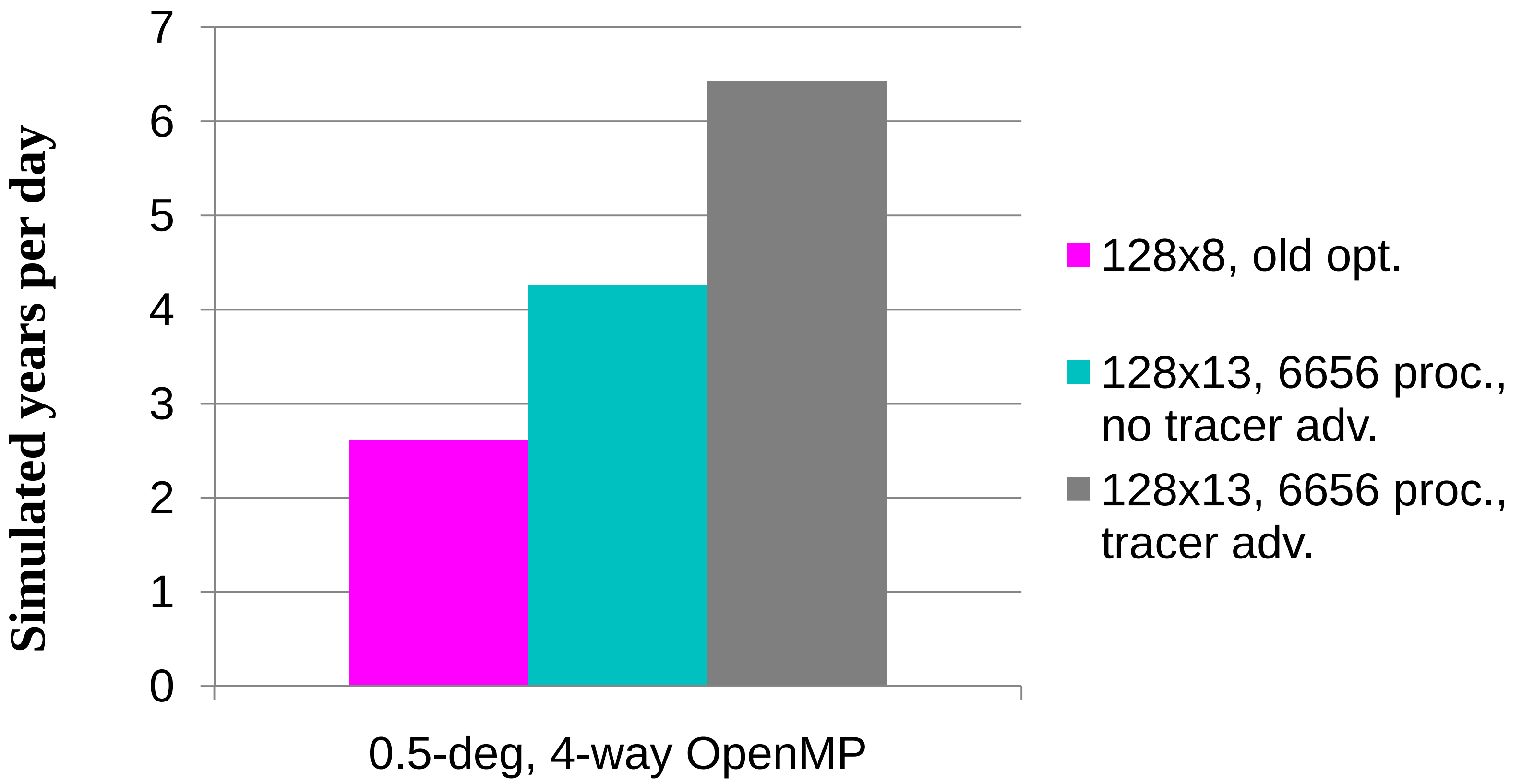
Improved throughput on Cray XT5 (vs OpenMP)



Fast reproducible distributed sums



Performance improvement with trop_mozart chemistry



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
- Run benchmark tests to understand costs of different physics and chemistry options
- 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 vectorization (e.g., Opteron SSE and 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 Atmospheric and Climate Research Division and the Office of Mathematical, Information, and Computational Sciences, 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-??????. 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.

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 compute time
  - — those particular tasks post receive requests to (primary) source tasks that themselves are posting receive requests
  - — the primary source tasks are themselves receiving send requests from their own (secondary) source tasks (particularly ones that are not target tasks)
  - — the resulting contention causes the delay
  - — solution is *handshaking*: delay send requests from secondary source tasks until primary source tasks are ready
  - — equal size decompositions are less likely to cause a problem since the secondary source send requests come naturally later in real time