

Comparison of XT3 and XT4 Scalability

Patrick H. Worley
Oak Ridge National Laboratory

CUG 2007
May 7-10, 2007
Red Lion Hotel
Seattle, WA



Acknowledgements

- Research sponsored by the Climate Change Research Division of the Office of Biological and Environmental Research and by the Office of Mathematical, Information, and Computational Sciences, both in the Office of Science, U.S. Department of Energy under Contract No. DE-AC05-00OR22725 with UT-Battelle, LLC.
- This research used resources (Cray XT3 and Cray XT4) of the National Center for Computational Sciences at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725 with UT-Battelle, LLC.
- These slides have been authored by a contractor of the U.S. Government under contract No. DE-AC05-00OR22725. 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.

Goals

- The (modest) end goal of this work is to determine how similar the XT3 and XT4 appear to a user, especially at scale. That is, we want to determine how the performance characteristics of the XT3 and XT4 differ, and what the implications are, if any, of these differences for performance optimization strategies for the two systems.
 - Why? Because ORNL has both an XT3 and an XT4 system.
- The (difficult) preliminary task is a performance characterization of both systems.

Cray XT3 vs. Cray XT4

- XT3 uses DDR-400 memory (peak 6.4 GB/s BW);
XT4 at ORNL uses DDR2-667 memory (peak 10.6 GB/s).
=> XT4 memory performance potentially 60% better than XT3.
- XT3 uses the Cray SeaStar Network Interface Controller (NIC) (2.2 GB/s peak injection BW, 4 GB/s sustained network BW);
XT4 uses SeaStar2 NIC (4 GB/s peak injection BW, 6 GB/s sustained network BW)
=> XT4 MPI performance potentially almost twice that of XT3.

Questions:

Do users need to retune their codes when moving between the XT3 and XT4? What performance differences should they expect when running on the XT4 as compared to the XT3?

Methodology

Comparing XT3 and XT4 in areas of

1. Kernel and application performance
2. Performance sensitivities:
 - a) single core (SN) vs. dual core (VN)
 - b) compiler optimization options
 - c) runtime options, e.g. `-small_pages` and environment variables
 - d) MPI protocols and collectives

using a heterogeneous XT3/XT4 system (Jaguar)

- 5212 XT3 compute nodes (2.6GHz dual core Opteron)
- 6296 XT4 compute nodes (2.6GHz dual core Opteron)

sited in the National Center for Computational Sciences at Oak Ridge National Laboratory, running in XT3-only or XT4-only partitions when determining performance characteristics.

Caveats

- Version 6.1.6 of Portland Group compilers
- Version 1.5.31 of the Cray Programming Environment

Have already heard this week of features and options in newer versions of the system software that will change the quantitative results described here (but, perhaps, not the qualitative results). Also, have not been exhaustive in exercising the current set of options that can affect performance. Your mileage may vary.

Optimization Options

Compiler flags:

default, -O2, -O3, -fast, -fastsse, -Mipa=fast, and combinations of these

Page size:

2 MB pages (default), 4 KB pages (-small_pages)

Mode:

One user process assigned to each node (SN), two user processes assigned to each node (VN)

Process placement:

wrap placement (default), SMP-style placement (MPICH_RANK_REORDER_METHOD == 1)

MPI options:

MPICH_COLL_OPT_ON (undefined, defined),
MPICH_FAST_MEMCPY (undefined, defined)

Conclusions

The performance characteristics of the Cray XT3 and XT4 architectures are qualitatively identical, with the same optimization strategies appropriate for both systems. For more details see the paper.

Outline of Talk

(guaranteed not to exceed)

Kernel Benchmarks

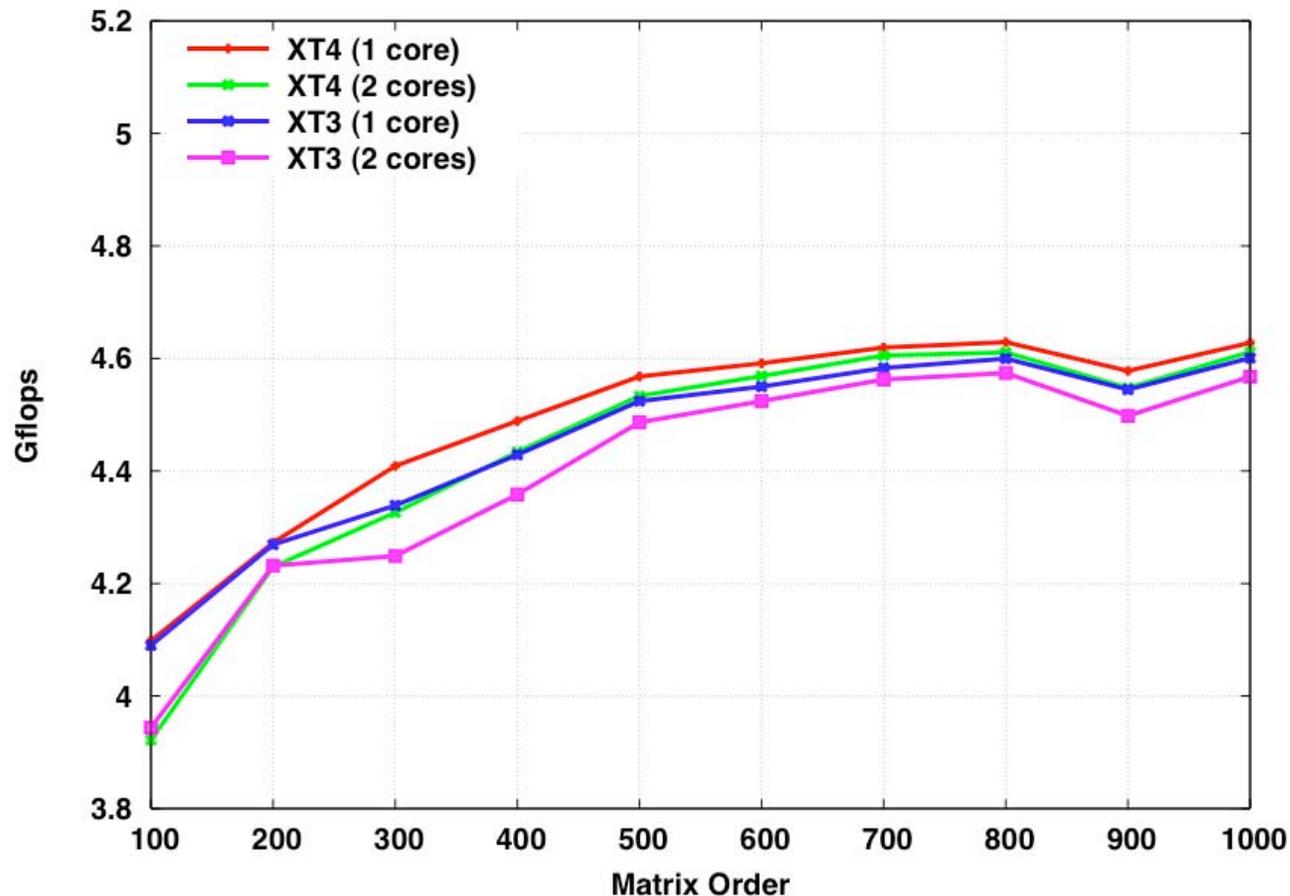
(compare with Monday talk on HPCC results)

- 1) Computation Benchmarks:
 - a) DGEMM
 - b) Parallel Spectral Transform Shallow Water Model (PSTSWM)
- 2) Communication Benchmarks:
 - a) COMMTEST
 - b) HALO

Application Benchmarks

- 3) POP (Parallel Ocean Program)
- 4) CAM (Community Atmosphere Model)
 - a) Spectral Eulerian Dynamics
 - b) Finite Volume Dynamics

Matrix Multiply Benchmark (DGEMM)



Evaluated performance of libsci routine for matrix multiply. Achieved 88% of peak, with small performance differences attributable to differences in memory performance (XT4 better than XT3; SN better than VN). Using `-small_pages` decreased performance slightly.

PSTSWM Benchmark

The Parallel Spectral Transform Shallow Water Model represents an important computational kernel in spectral global atmospheric models. As 99% of the floating-point operations are multiply or add, it runs well on systems optimized for these operations. PSTSWM exhibits little reuse of operands as it sweeps through the field arrays; thus it exercises the memory subsystem as the problem size is scaled and can be used to evaluate the impact of memory contention in multi-core nodes. (“STREAM analog”)

PSTSWM Experiments

These experiments examine serial performance, both using one core and running the serial benchmark on both cores simultaneously.

Performance is measured for a range of horizontal problems resolutions for 1 to 88 vertical levels.

Horizontal Resolutions

T5: 8 x 16

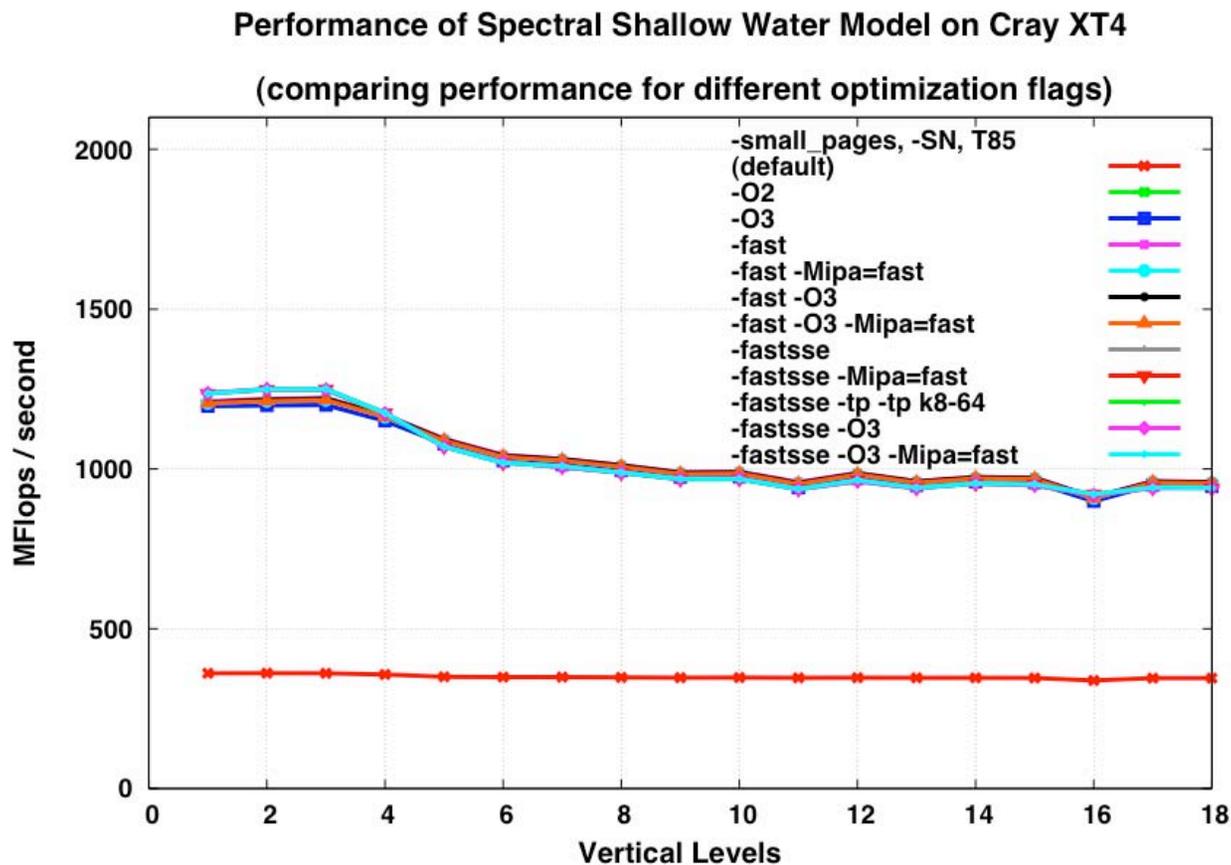
T10: 16 x 32

T21: 32 x 64

T42: 64 x 128

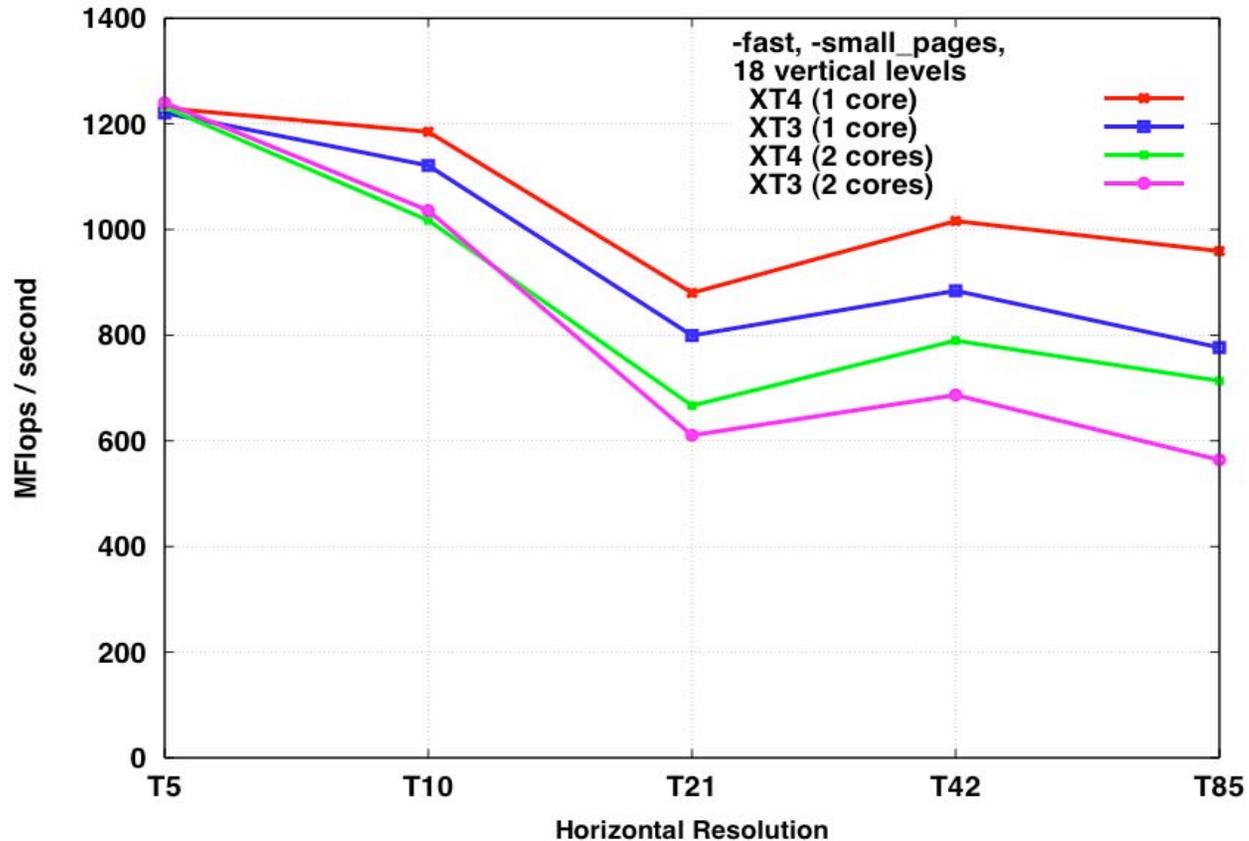
T85: 128 x 256

PSTSWM Compiler Comparisons on XT4



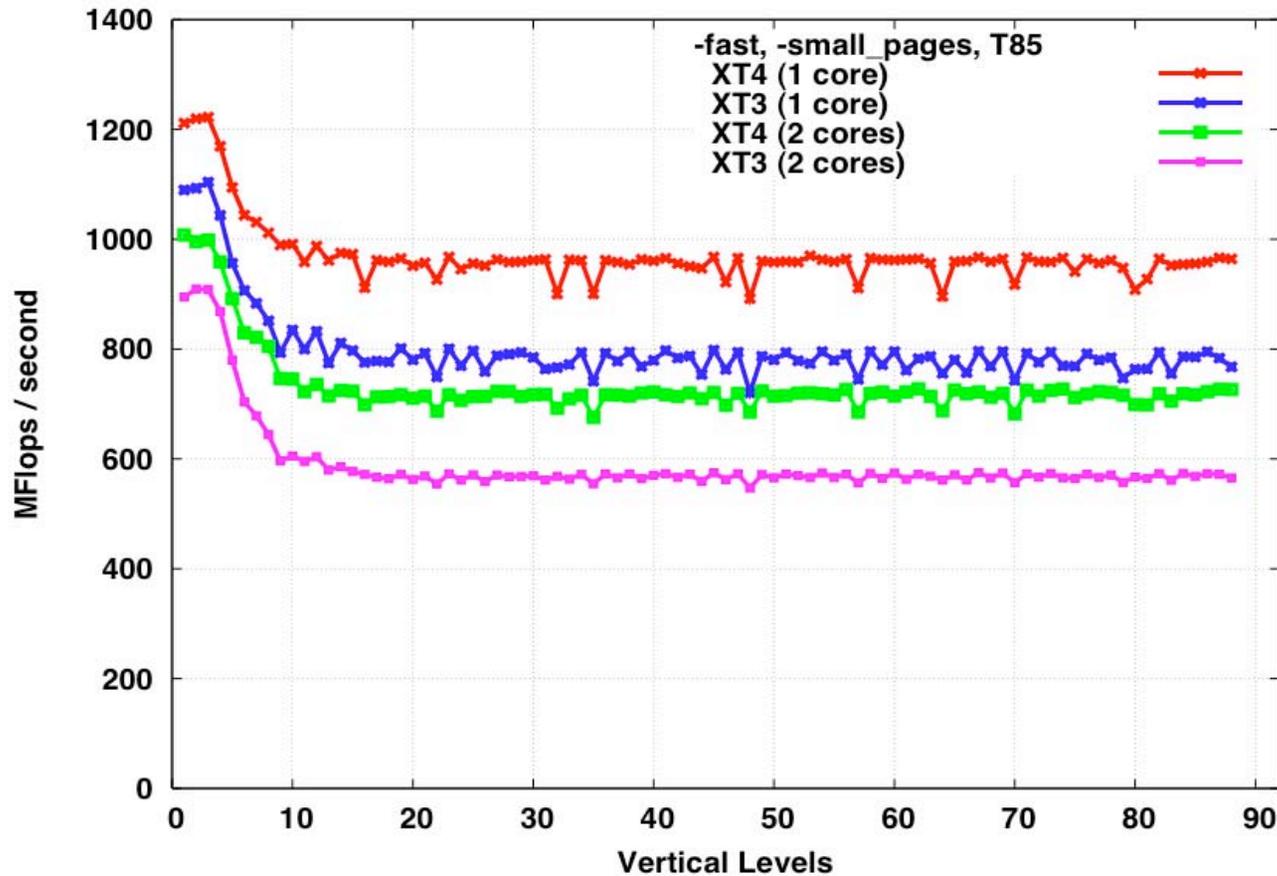
Comparing performance of different optimization levels for T85. For this code, -fast is as good as anything else. -small_pages also did not affect performance significantly. Similar results hold for VN mode and on the XT3.

PSTSWM Performance (varying horizontal resolution)



The XT4 is faster than the XT3; single core is faster than dual core; the computation rate (generally) decreases with horizontal resolution.

PSTSWM Performance (varying vertical resolution)



The XT4 is faster than the XT3; single core is faster than dual core; the computation rate decreases with vertical resolution during initial increase, after which it becomes relatively constant.

Computation Kernels Summary

- 1) The XT3 and XT4 achieved nearly identical peak computation rate.
- 2) Qualitatively, the XT3 and XT4 performance characteristics are identical with respect optimal compiler optimizations, performance degradation due to high memory traffic, and performance degradation due to contention for memory when running on both cores.
- 3) When memory performance constrains computation rate,
 - a) XT4 is faster than XT3;
 - b) Single core performance is superior to dual core performance (on a per core basis).

Everything behaved as advertised.

COMMTEST Benchmark

- COMMTEST is a suite of codes that measures the performance of MPI interprocessor communication. In particular, COMMTEST evaluates the impact of communication protocol, packet size, and total message length in a number of “common usage” scenarios. (However, it does not include persistent MPI point-to-point commands among the protocols examined.)
- The benchmark was compiled with `-fast` and run with `-small_pages` .
- The benchmark was run with
`setenv MPICH_RANK_REORDER_METHOD 1`
so that processes have the expected order in the experiments.
- The benchmark was run both with and without
`setenv MPICH_FAST_MEMCPY 1`
but the results were unchanged.

COMMTEST Experiments

i-j

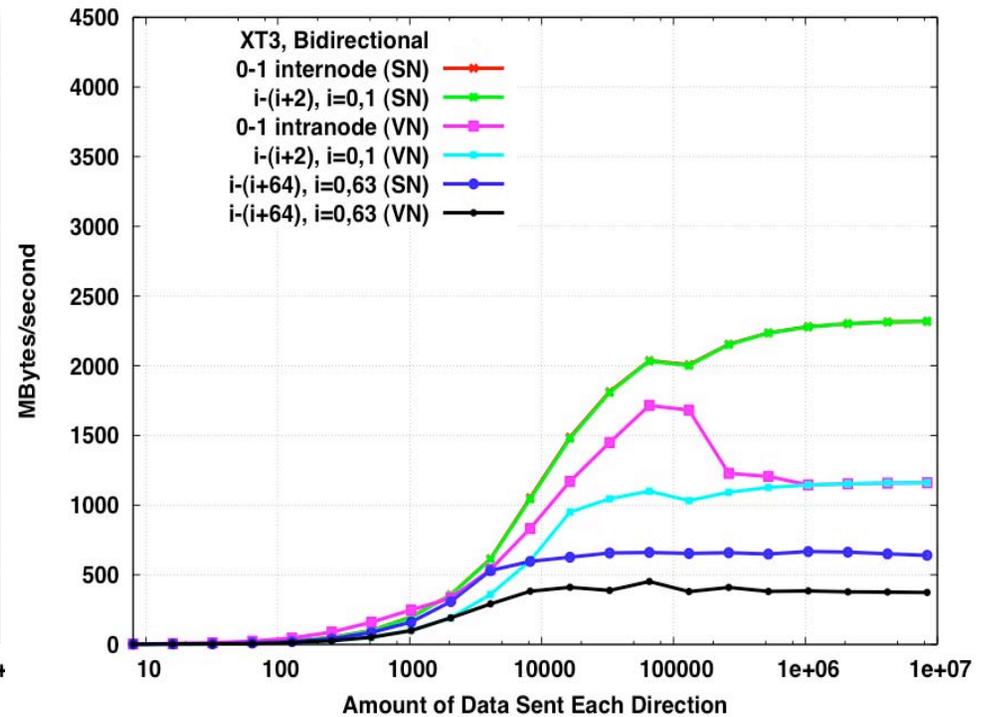
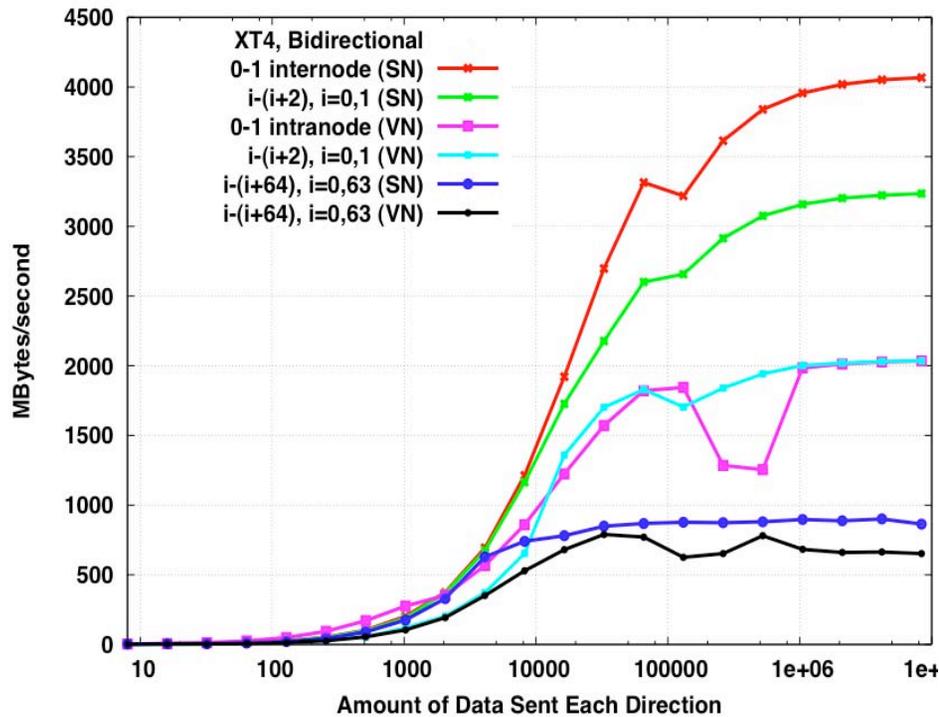
processor i swaps data with processor j . Depending on i and j , this can be within a node or between nodes.

i-(i+j); $i=1, \dots, n$; $n < j$

n processor pairs $(i, i+j)$ swap data simultaneously. Depending on j , this will be within a node or between nodes (or both). Minimum per pair performance is reported.

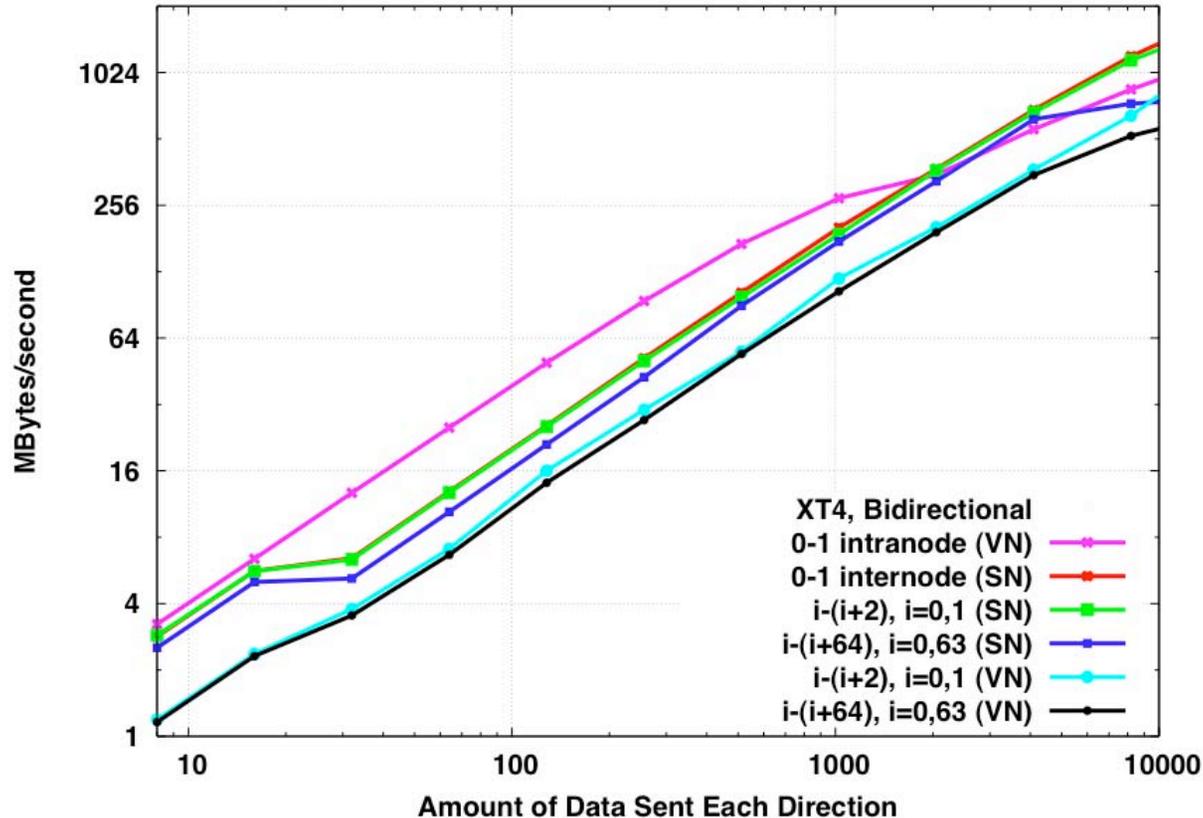
Note: experiments were not run in dedicated mode. Nodes were usually, but not always, physically “contiguous”, but part of torus used was not controlled as part of the experiment.

Bidirectional BW: XT3 vs. XT4



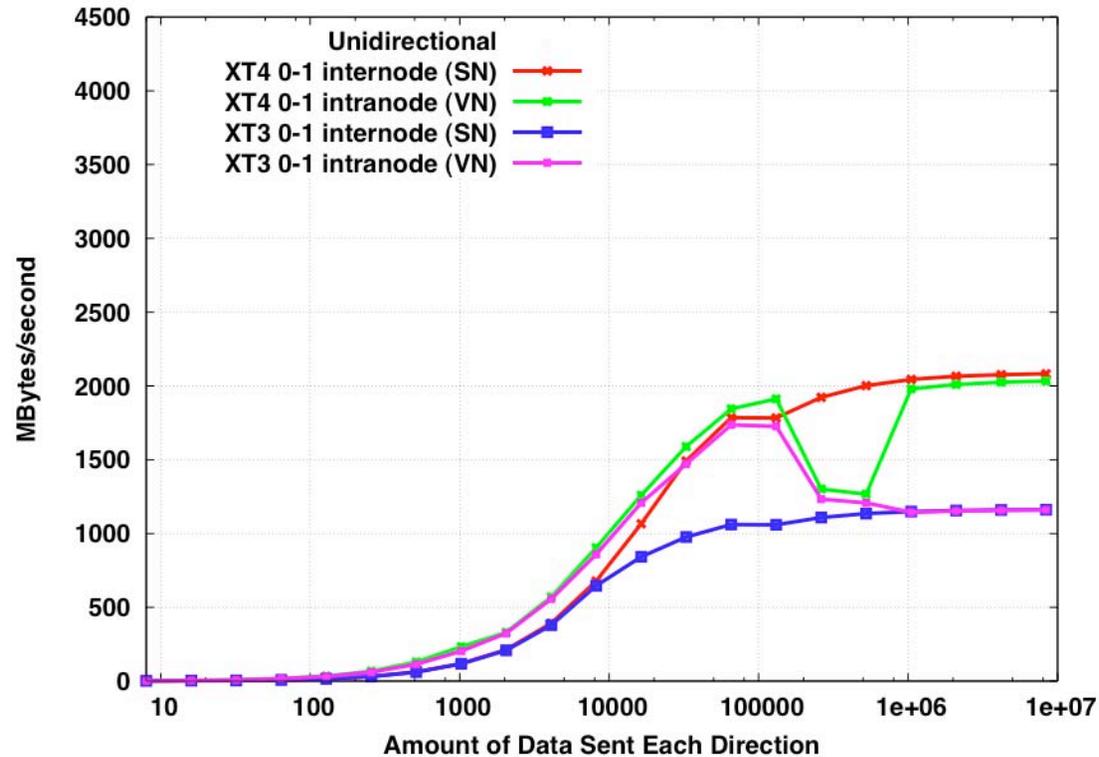
Bidirectional bandwidth on the XT4 is as much as twice that on the XT3, as advertised. The improvement is less in the situations exhibiting network link contention. 2-pair SN swap shows link contention on XT4, but not on XT3. Otherwise, performance is qualitatively similar on the XT3 and the XT4 for large messages.

Bidirectional BW: small messages



Intranode latency (0-1 VN) is smaller than internode latency. Two cores per node communicating off node exhibits twice the latency of one core per node communicating. XT4 small message performance is almost identical to that of the XT3.

Unidirectional BW: XT3 vs. XT4



Unidirectional internode bandwidth is half that of bidirectional. Unidirectional intranode bandwidth is the same as bidirectional. Unidirectional bandwidth on the XT4 is (again) as much as twice that on the XT3, but is otherwise qualitatively similar for large messages. For small messages, XT4 performance is identical to XT3 performance.

Other COMMTEST Results

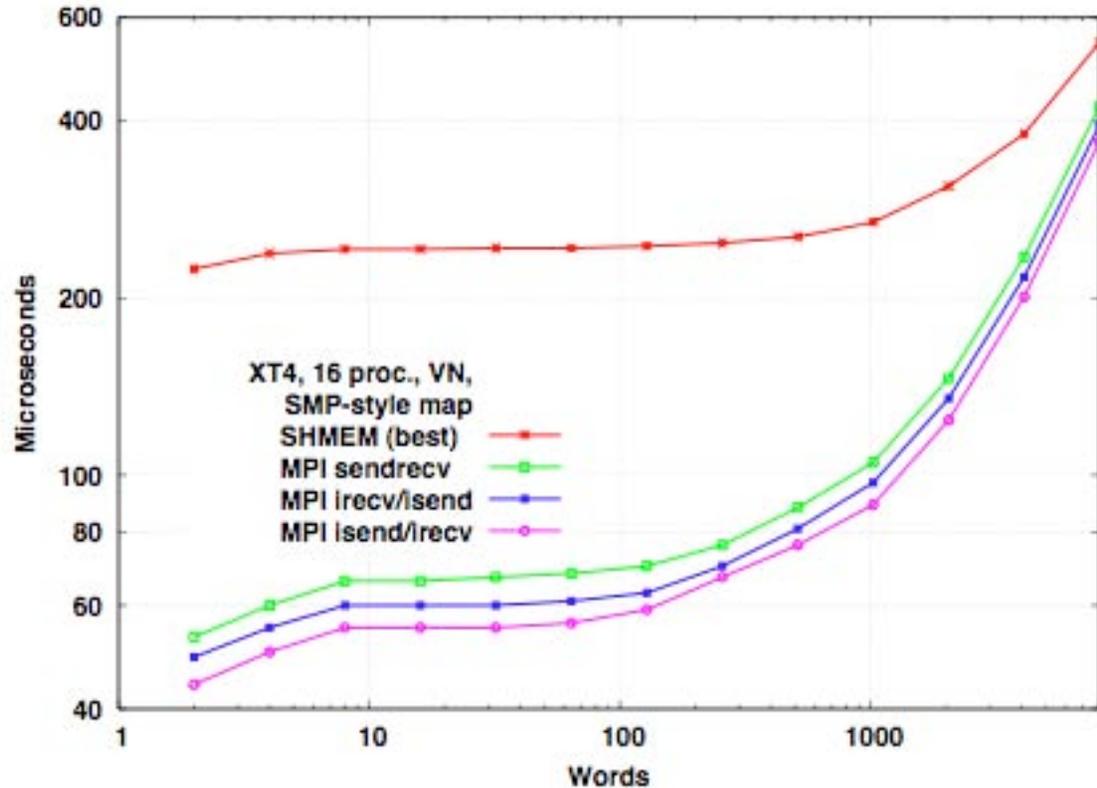
MPI 2-sided communication protocols that achieve, or nearly achieve, reported performance:

- For a single pair swap (0-1):
 - For large message sizes, performance is relatively insensitive to choice of protocol.
 - For small message sizes, the best protocol is MPI_Isend/MPI_Recv for SN mode and MPI_Sendrecv for VN mode.
- For 64-pair simultaneous swap:
 - For large message sizes, performance is optimized by preposting receives and using ready sends (for both SN and VN modes).
 - For small message sizes, MPI_Sendrecv is competitive for both SN and VN modes.

HALO Benchmark

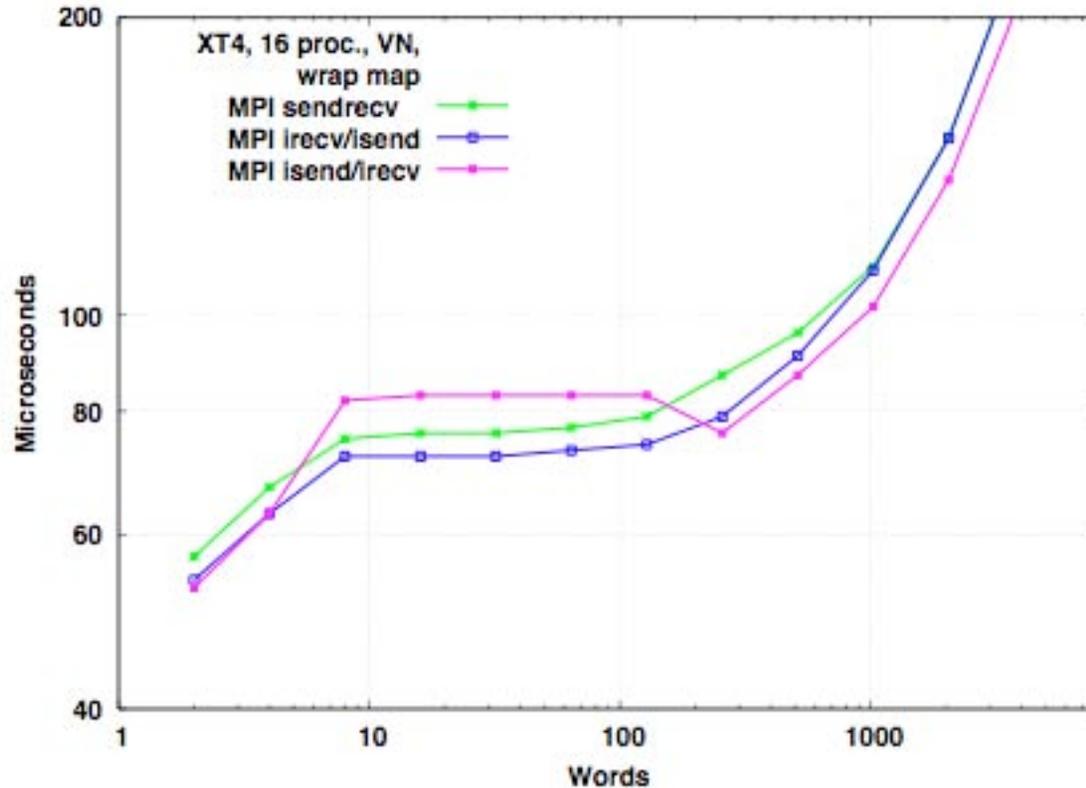
- Alan Wallcraft's HALO benchmark is a suite of codes that measures the performance of 2D halo updates as found in 2D domain decompositions of, for example, finite difference ocean models. Implementations exist for multiple message layers, and for multiple implementations for a given layer.
- We used HALO on 16 processors to compare MPI two-sided communication protocols and to compare MPI with SHMEM. We also examined the performance impact of process placement.
- The benchmark was compiled with `-fast`. Part of the measured time is copying buffers, and HALO experiments with default optimization are much slower than with `-fast` optimization. Experiments were run with both large pages and small pages, and performance was unchanged.

HALO Protocol Comparison on XT4



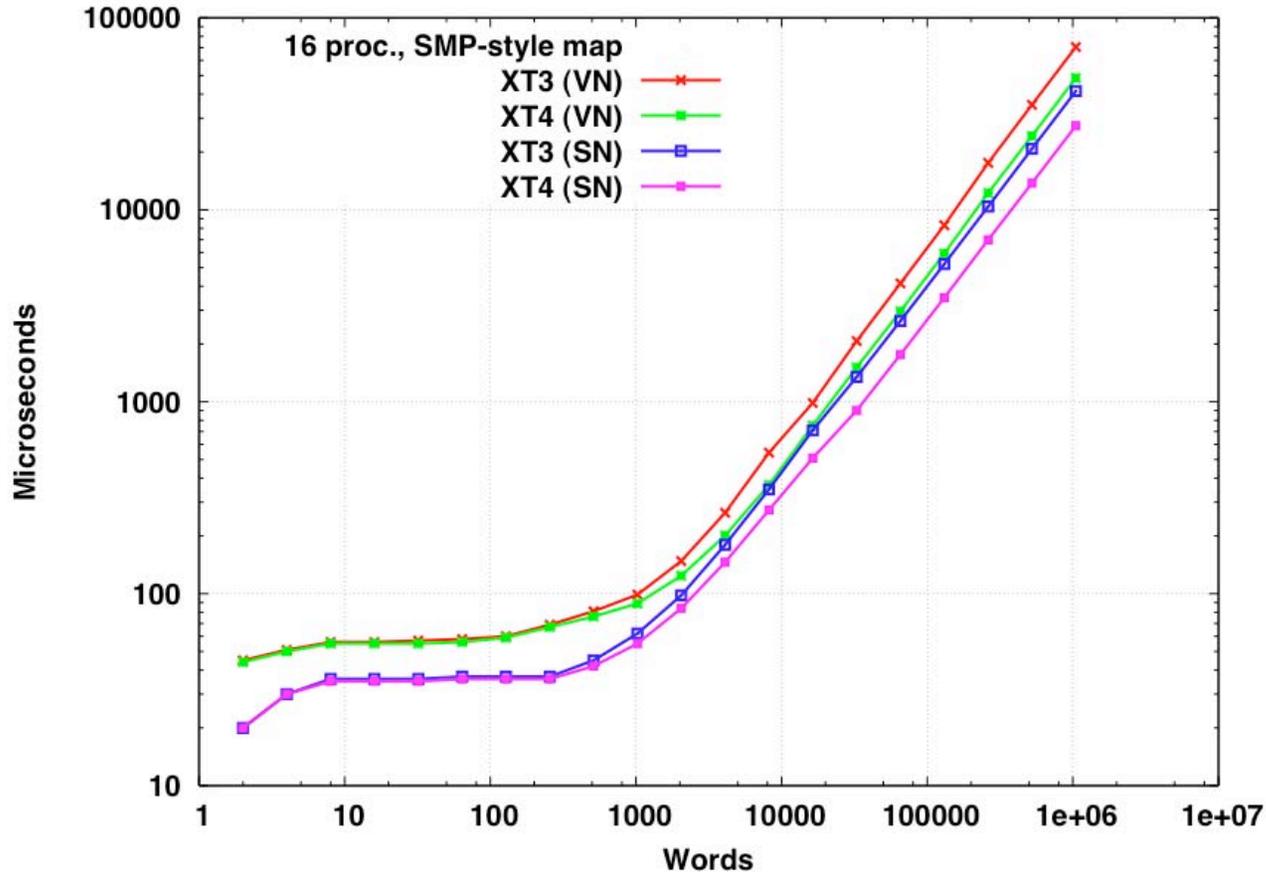
For small haloes and SMP-style process placement, the MPI_Isend/MPI_Irecv communication protocol achieved the best performance. For large haloes (not shown here), the examined MPI protocols achieved essentially the same performance. Performance for SHMEM was much worse than for MPI. Results were qualitatively the same in SN mode and on the XT3.

HALO Protocol Comparison on XT4



For small haloes and wrap process placement, the MPI_Isend/MPI_Irecv communication protocol is sometimes the worst performer. Wrap process placement degrades performance compared the SMP-style placement for all protocols and for all halo sizes. Results were qualitatively the same in SN mode and on the XT3.

HALO Benchmark: XT3 vs. XT4



For small haloes, XT4 performance is identical to XT3 performance, and SN performance is twice that of VN. For large haloes, XT4 performance is twice that of XT3, and SN performance is (still) twice that of VN.

Communication Kernels Summary

- 1) The XT3 and XT4 exhibit the same small message latency.
- 2) The XT4 achieves twice the large message bandwidth as the XT3 when network contention does not limit performance.
- 3) SN mode achieves twice the large message interprocessor bandwidth as VN mode (but the same internode bandwidth).
- 4) SN mode exhibits half the latency of VN mode.
- 5) Qualitatively, the XT3 and XT4 performance characteristics are identical with respect to optimal communication protocols, impact of process placement, and impact of link contention.

Everything behaved as expected, except for magnitude of difference between SN and VN latency,

Parallel Ocean Program (POP)

- Developed at Los Alamos National Laboratory. Used for high resolution studies and as the ocean component in the Community Climate System Model.
- Two primary computational phases:
 - Baroclinic: 3D with limited nearest-neighbor communication; scales well.
 - Barotropic: dominated by solution of 2D implicit system using conjugate gradient solves; scales poorly.
- Domain decomposition determined by grid size and 2D virtual processor grid.

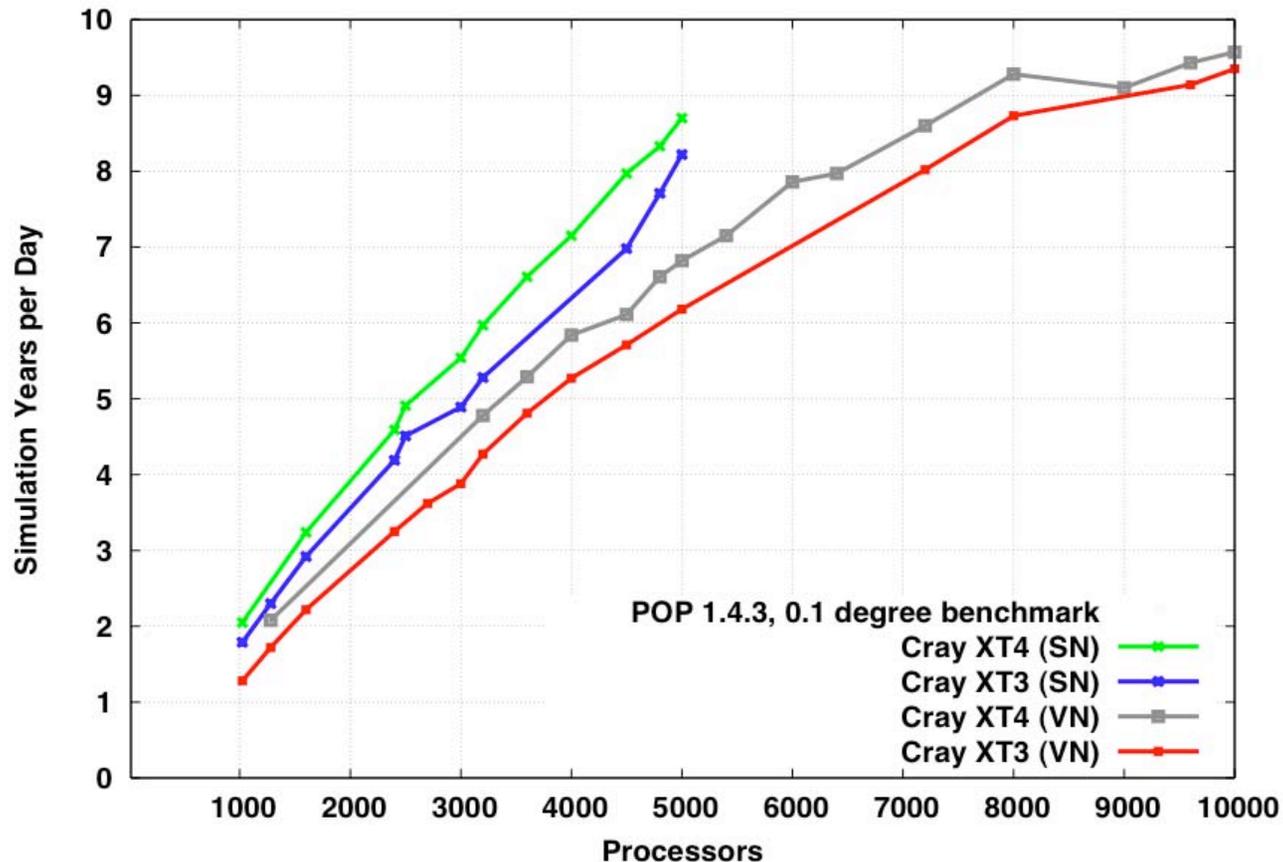
POP Experiment Particulars

- Los Alamos National Laboratory version of POP1.4.3 with a few additional parallel algorithm tuning options (due to Dr. Yoshida of CRIEPI).
 - Note: This is the “original” POP benchmark. The current production version of POP is version 2.0.1, and it is the focus of current optimization work. Version 1.4.3 is being used to evaluate machine performance, not to evaluate the performance of POP.
- One fixed size benchmark problems
 - Tenth degree horizontal grid of size 3600x2400x40 using internally generated horizontal grid
- Results for a given processor count are the best observed over all applicable processor grids.

POP Experiment Particulars (cont.)

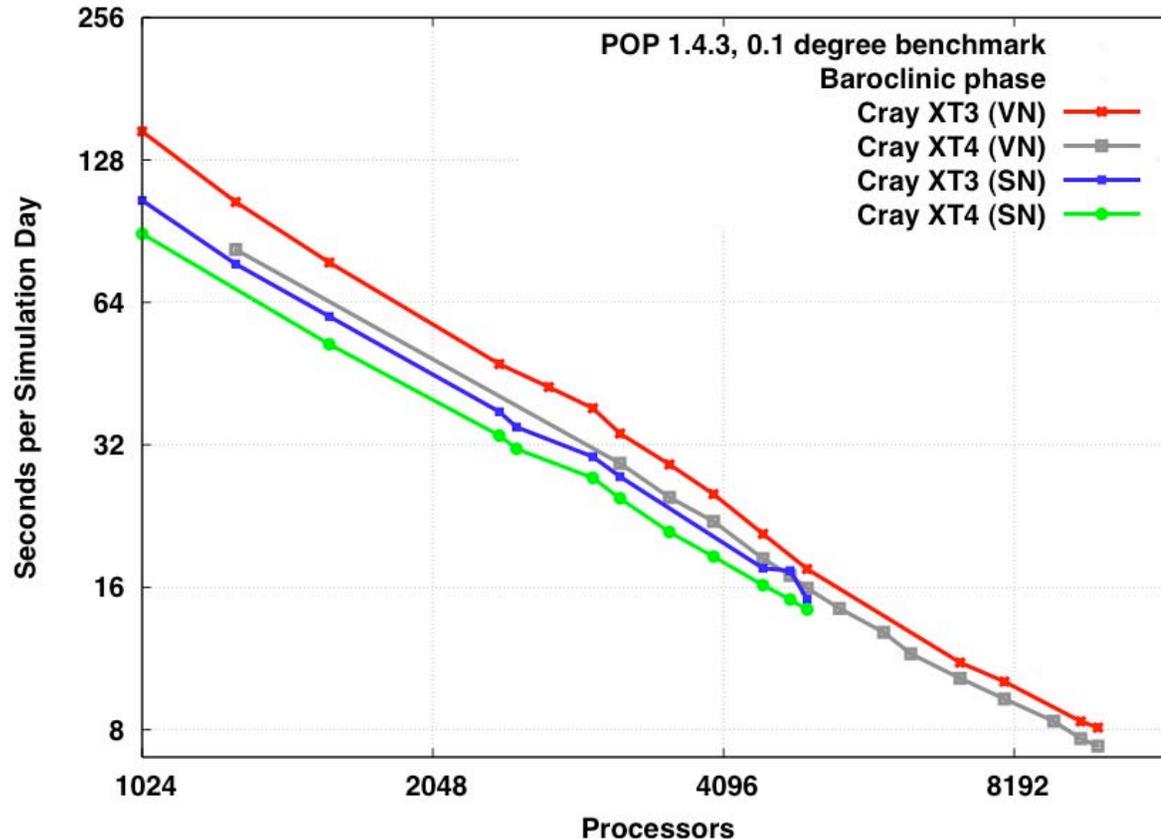
- Compiled with `-Kieee -O3 -fastsse -tp k8-64` (but performance not much different when compiling with `-Kieee -fast`).
- Ran with both small and large pages. Performance with small pages was slightly better.
- Experimented with setting `MPI_COLL_OPT_ON`, but observed no performance advantage.

POP Benchmark Performance



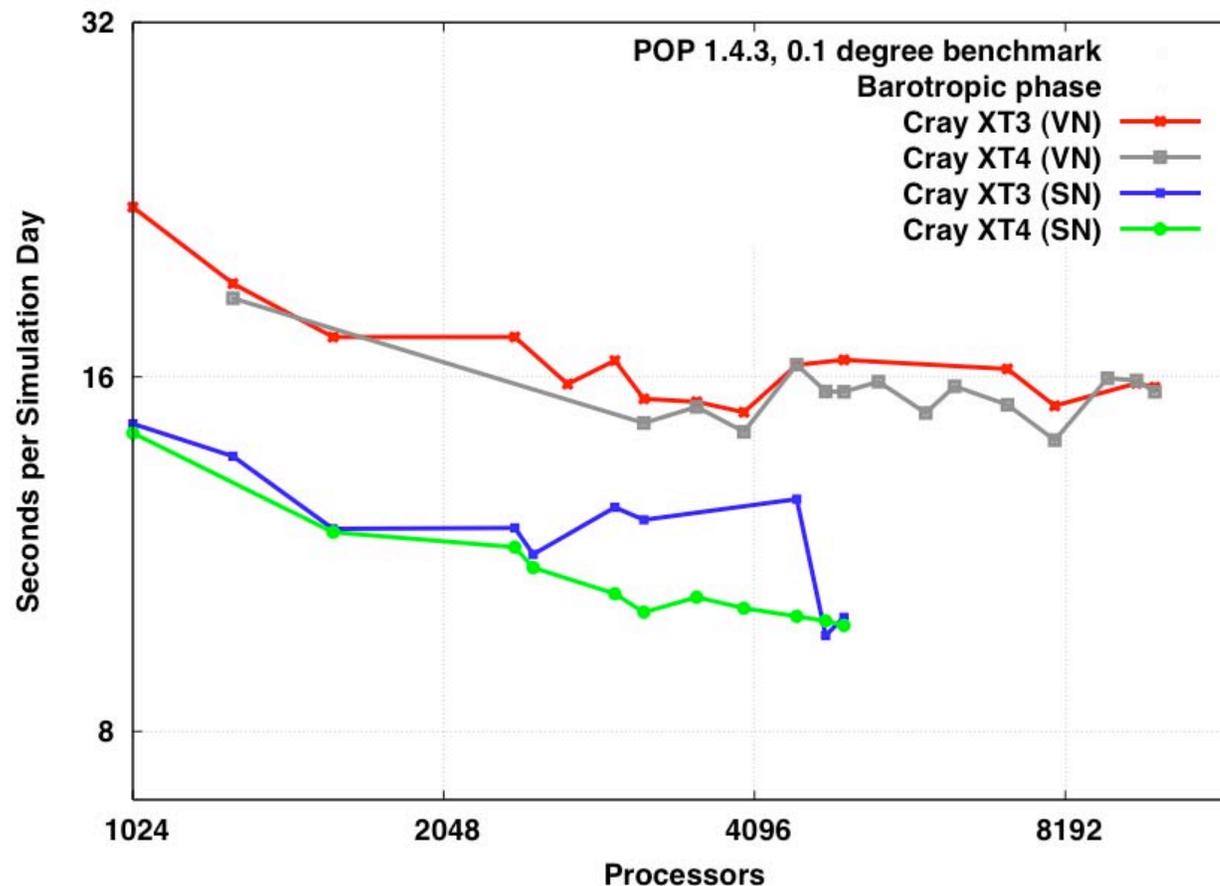
The XT4 faster is than the XT3. SN mode is *much* faster than VN mode for the same number of processors, and is not much slower for the same number of compute nodes. The XT3 and XT4 performance characteristics are similar.

POP Baroclinic Phase



The baroclinic phase is scaling as expected. As with the whole code, the XT4 is faster than the XT3, and SN mode is faster than VN mode (for the same processor count). However, the degree to which SN mode is faster than VN mode is much smaller, and VN is much faster for the same number of compute nodes.

POP Barotropic Phase



The barotropic phase does not scale at all beyond 4096 processors in VN mode, and scaling is minimal in SN mode. (At least execution time hasn't started increasing.) XT3 and XT4 performance are almost identical, modulo performance perturbations.

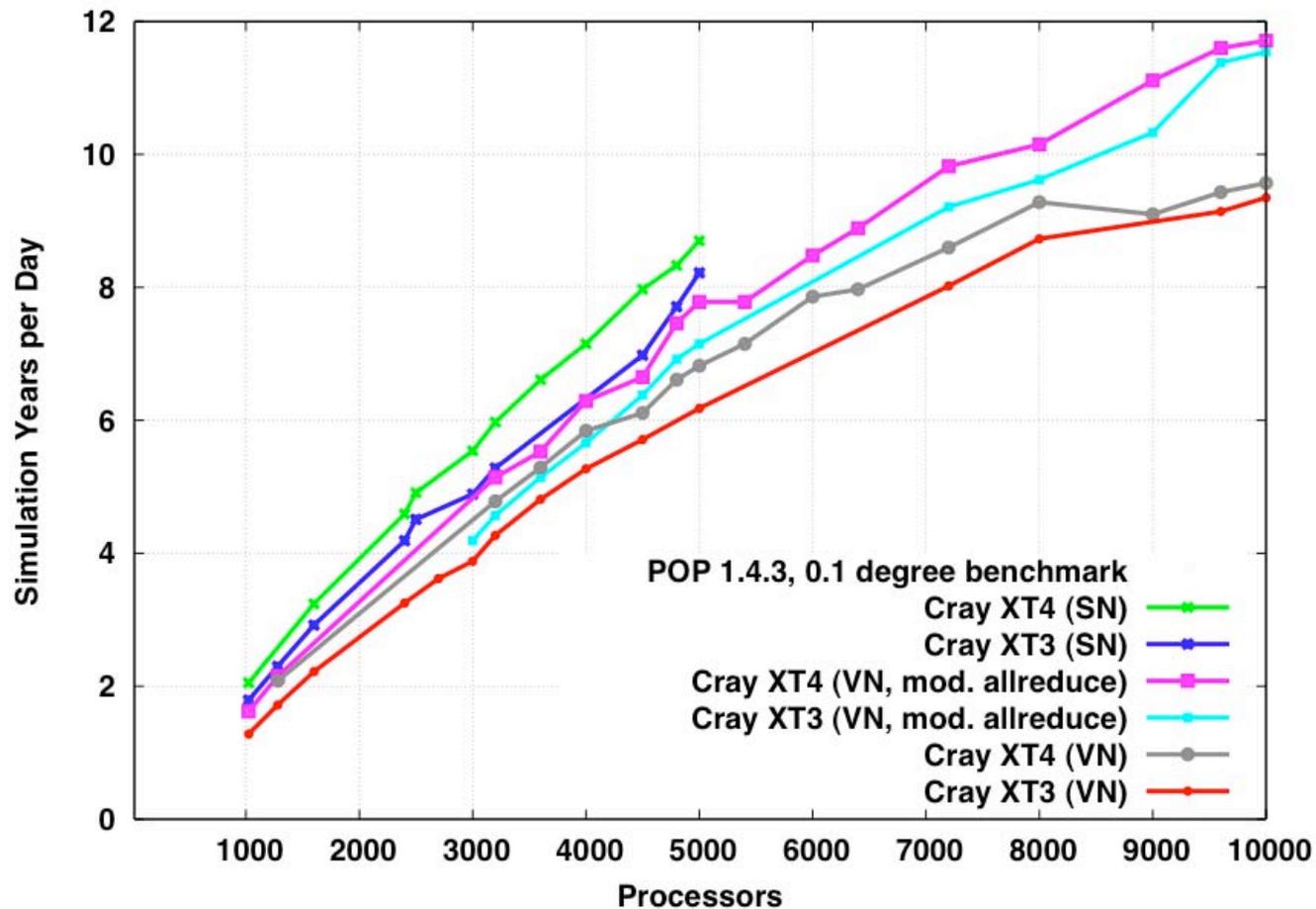
Modified POP Benchmark

- Problem: Poor VN mode performance in barotropic phase. Kernel results suggest that the problem is due to poor VN mode latency compared to SN mode.
- Solution: Replace MPI_Allreduce over MPI_COMM_WORLD with MPI_Allreduce over a subcommunicator containing only “core 0” processors.
- Implementation:
 - Core 1 send local sum to core 0; core 0 adds this to its local sum
 - Call MPI_Allreduce on “core 0” subcommunicator
 - Core 0 sends result to core 1.

Note: used SMP-style placement to simplify algorithm development.

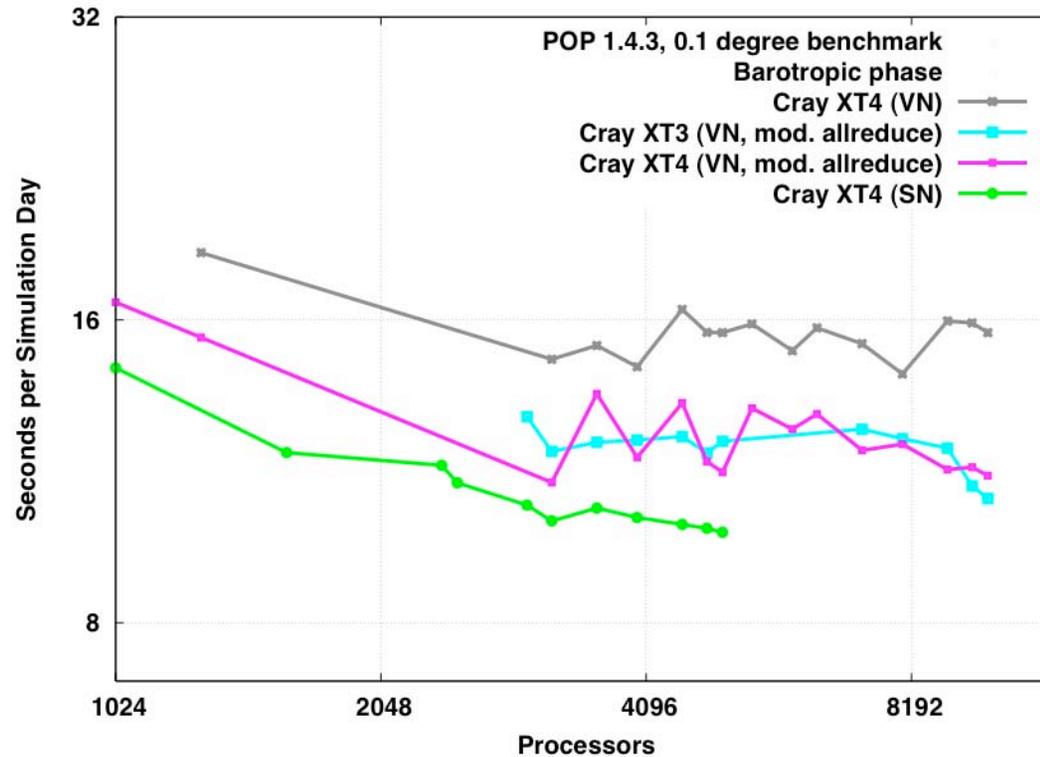
*Short term fix? Should be implemented in MPI collectives?
Problem will disappear with Compute Node Linux?*

POP Benchmark Performance



Modified MPI_Allreduce improves POP performance significantly, especially at scale. Improvement is the same for both the XT3 and the XT4.

POP Phase Analysis



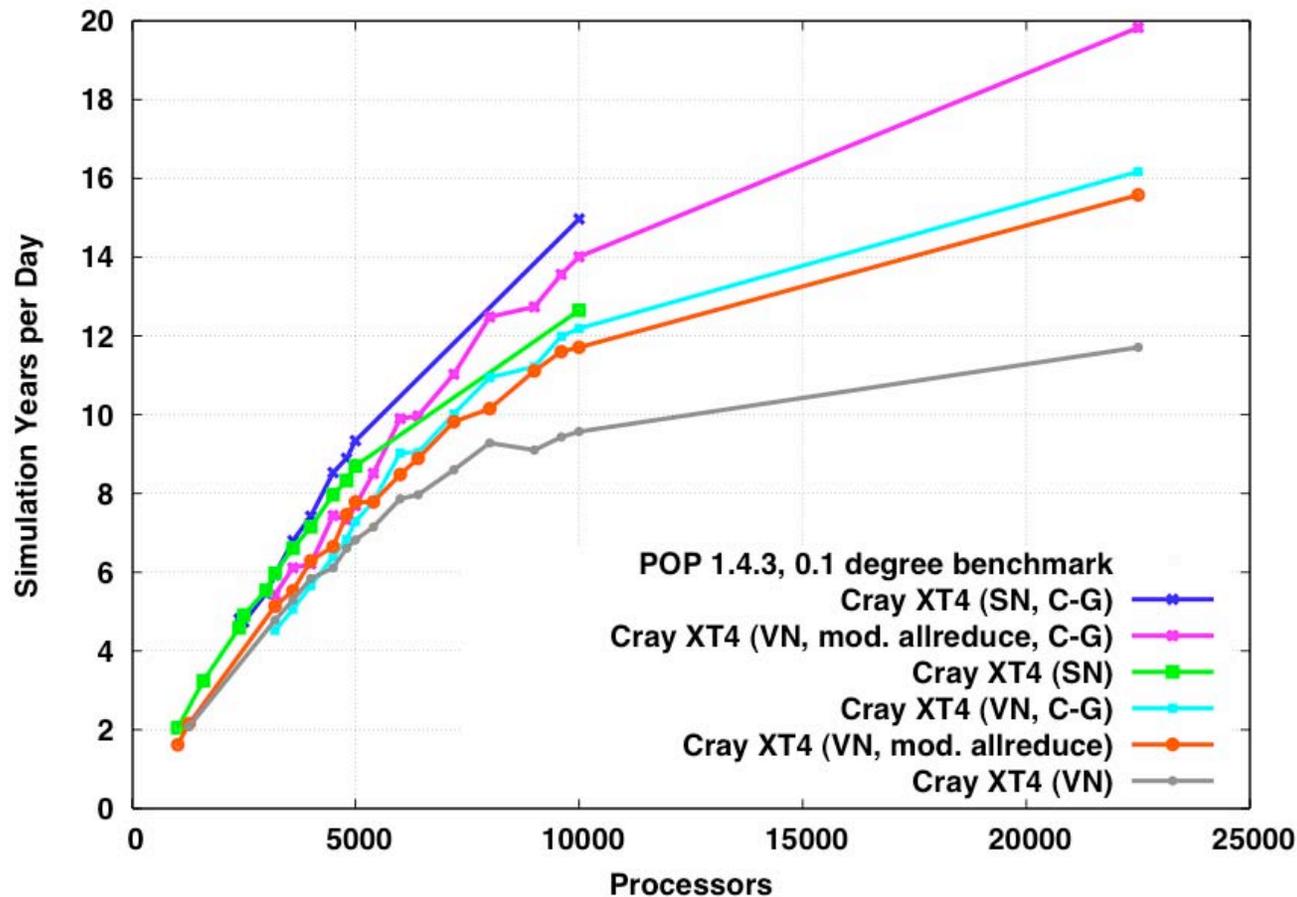
The modified MPI_Allreduce improves performance of the barotropic phase, but performance is still sensitive to perturbations. The barotropic phase also includes a halo update. Typically the MPI_Allreduce dominates performance, but this needs to be re-examined.

Modified POP Benchmark II

- Problem: Barotropic performance limiting performance scalability.
- Solution: Decrease number of MPI_Allreduce calls in barotropic phase by using Chronopoulos-Gear (C-G) variant of the Conjugate-Gradient solver.
- Implementation:
 - Back ported C-G code from later versions of POP.

Note: POP production runs are already using C-G. However, they are not (yet) using the modified MPI_Allreduce algorithm. POP 2.0.1 does include an option to run the barotropic phase on a smaller number of processors. If this subset is defined to be one processor per node, the option may provide another approach to eliminating the VN mode performance degradation.

POP Benchmark Performance



Combination of modified MPI_Allreduce algorithm and C-G variant of conjugate gradient improved performance significantly. In particular, SN mode performance is now only slightly better than VN mode for large processor counts, and VN mode is much faster as a function of compute nodes.

POP Benchmark Summary

- 1) Cost of MPI_Allreduce, especially in VN mode, limits POP scalability.
- 2) Replacing MPI_Allreduce with an SN mode implementation alleviates problem.
- 3) Reducing the number of calls to MPI_Allreduce further improves performance.
 - *Achieved highest reported POP performance (of any version) for this benchmark. Note that results for largest processor counts included both XT3 and XT4 compute nodes.*
- 4) XT3 and XT4 results are qualitatively similar, with the XT4 being slightly faster.

Community Atmosphere Model (CAM)

Atmospheric global circulation model

- Timestepping code with two primary phases per timestep
 - *Dynamics*: advances evolution equations for atmospheric flow
 - *Physics*: approximates subgrid phenomena, such as precipitation, clouds, radiation, turbulent mixing, ...
- Multiple options for dynamics:
 - Spectral Eulerian (EUL) dynamical core (*dycore*)
 - Spectral semi-Lagrangian (SLD) dycore
 - Finite-Volume semi-Lagrangian (FV) dycoreall using tensor product *longitude x latitude x vertical level* grid over the sphere, but not same grid, same placement of variables on grid, or same domain decomposition in parallel implementation.
- Separate data structures for dynamics and physics and explicit data movement between them each timestep (in a “coupler”)
- Developed at NCAR, with contributions from DOE and NASA

Spectral Eulerian Benchmark

- T85L26 grid: 128x256 latitude-longitude and 128x85 latitude-wavenumber horizontal grids, both with 26 vertical levels
- Dynamics limited to 1D decomposition in latitude, so can use at most 128 MPI processes.
- Physics supports arbitrary 2D horizontal decomposition, and is twice as expensive as the dynamics.
- Interprocessor communication in the dynamics includes remap between 1D latitude and 1D wavenumber domain decompositions.
- Interprocessor communication in “coupling” between the dynamics and the physics is a complete remap of domain decomposition when physics load balancing is enabled. Without physics load balancing, no communication is required.

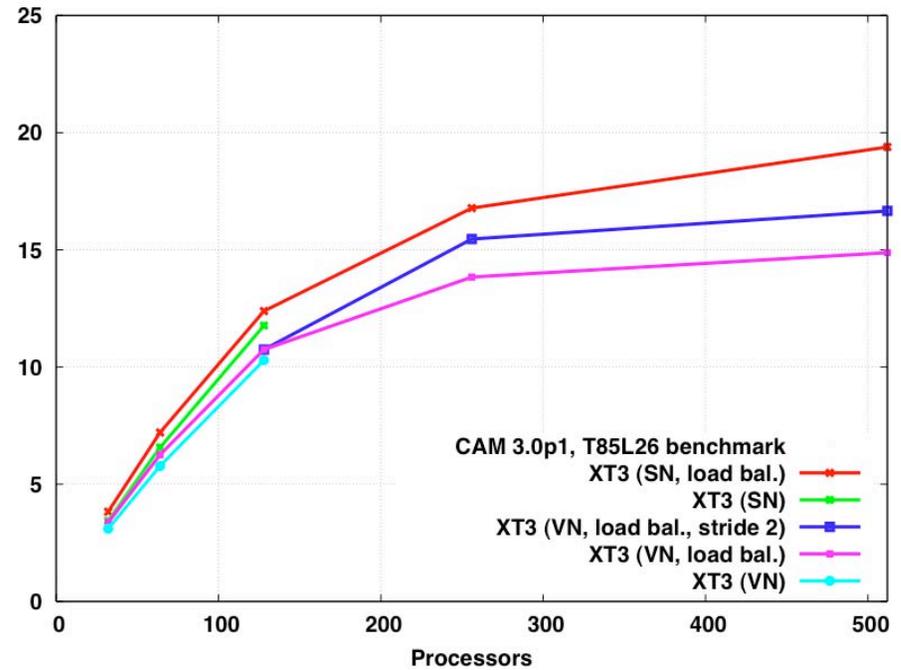
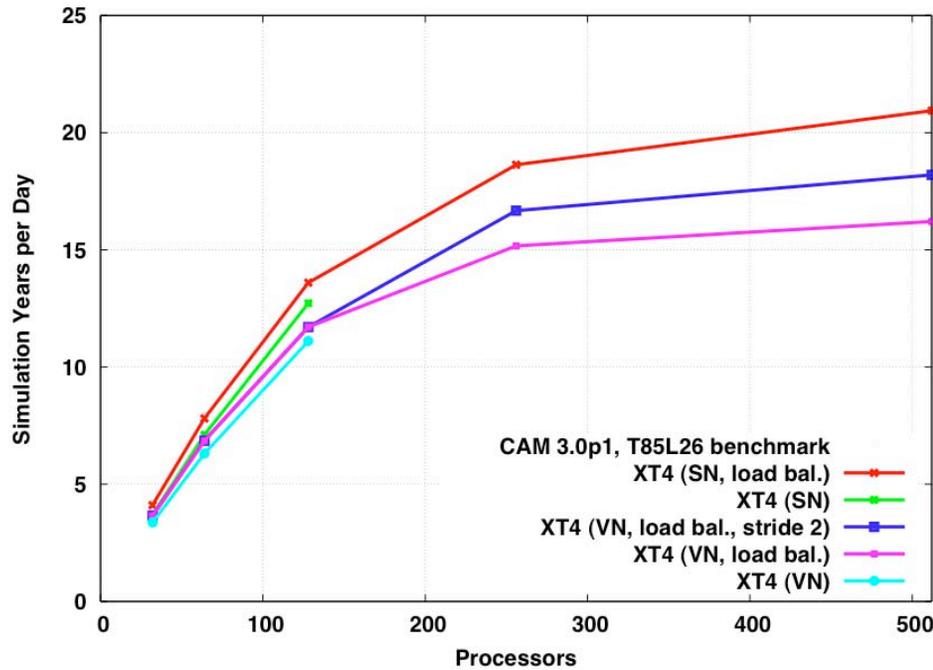
Spectral Eulerian Benchmark (cont.)

- Compiled with -fast; ran with both large and small pages. Using small pages was slightly faster.
- Ran with both MPICH_RANK_REORDER_METHOD 1 and 2, with and without MPI_COLL_OPTION, and with and without MPICH_FAST_MEMCPY. Performance differences were small and within the experimental variability.

Modified Spectral Eulerian Benchmark

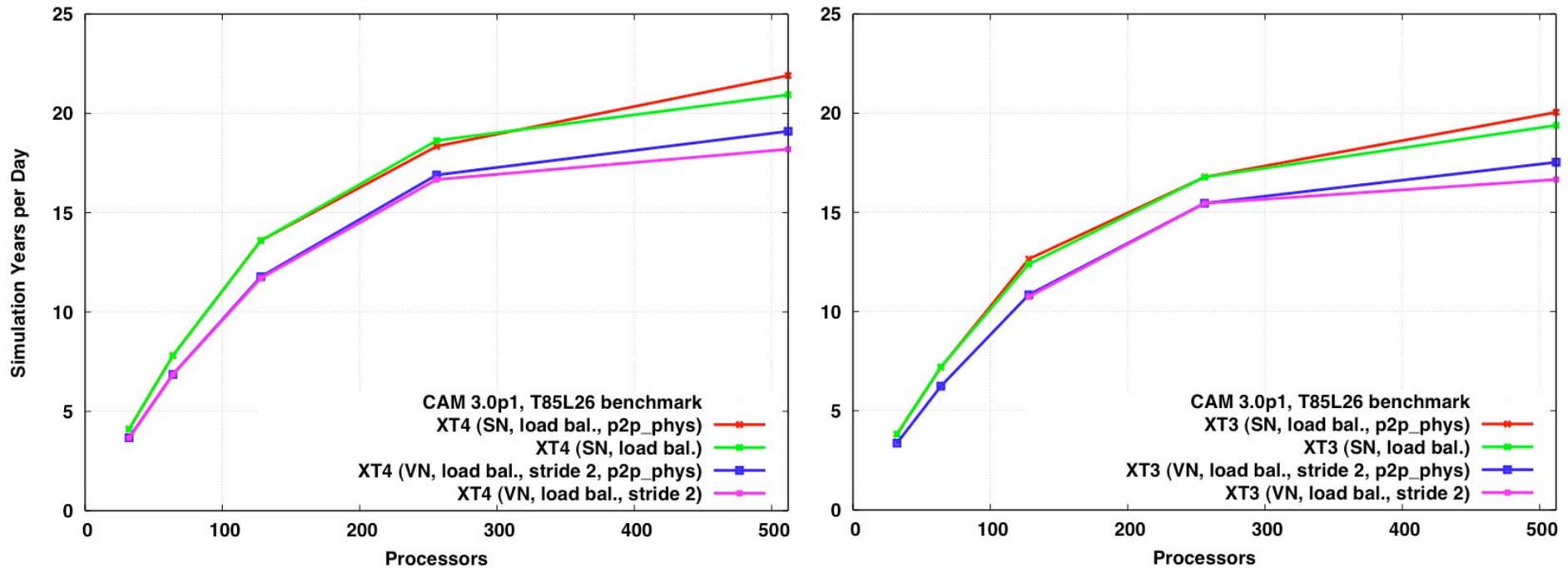
- Problem: Limited scalability due to lack of OpenMP support.
- Solution: Allow the dynamics and the physics to use different numbers of MPI processes.
- Implementation:
 - Automatically limit number of active processors in the dynamics, but allow as many as requested in the physics.
 - Allow user to specify stride (separation between active dynamics processors). So with stride 2 in VN mode using SMP-style placement, only one processor is active per node during the dynamics (equivalent to SN mode for the dynamics).
 - Allow user to choose between MPI collective and point-to-point implementations of domain decomposition remaps within the dynamics and in the dynamics-physics coupling. For point-to-point implementations, support 19 different MPI protocols.

Spectral Eulerian Performance



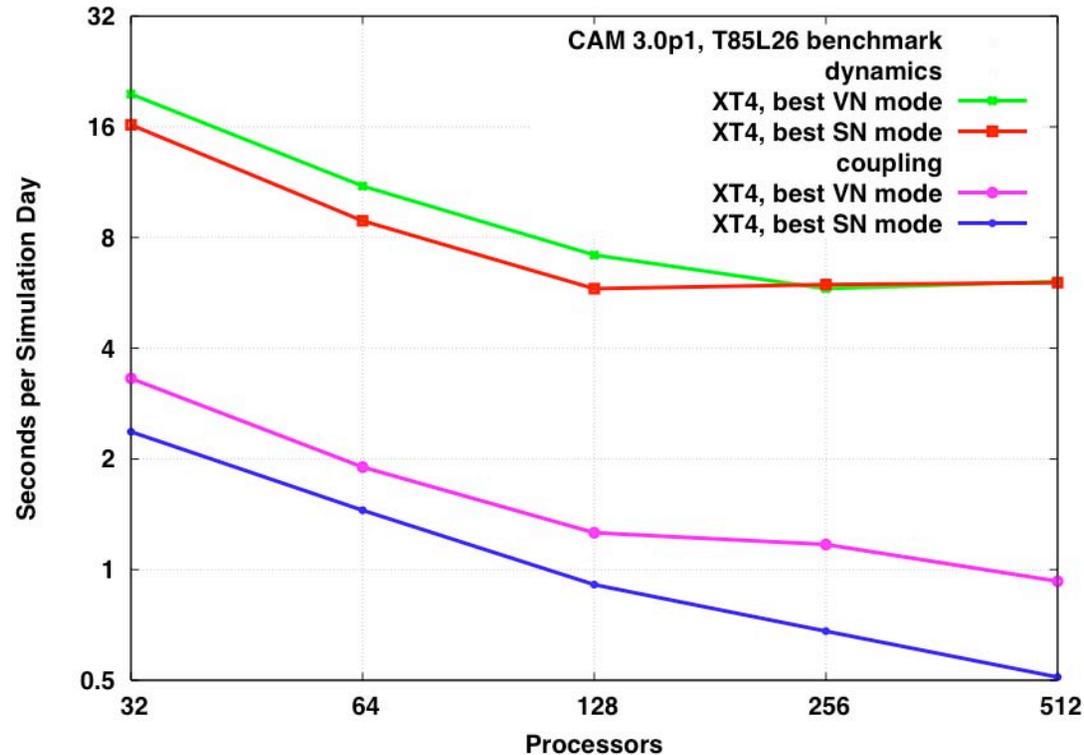
Using modified benchmark with MPI collective implementation, we were able to extend scalability out to 512 processors. SN mode is faster than VN mode, but running the dynamics in SN mode (stride 2) is faster than running in all VN mode when using more than 128 processors. Qualitatively, XT4 performance is identical to XT3 performance. Quantitatively, XT4 is faster than XT3 by approximately 10%.

Spectral Eulerian Performance



Using modified benchmark and comparing MPI collective implementation with best point-to-point implementation (MPI_Isend/MPI_Recv) for communication between the dynamics and the physics. For 512 processors, point-to-point is faster (and it is competitive for other processor counts). Conclusions same for the XT3 and the XT4.

Spectral Eulerian Phase Analysis

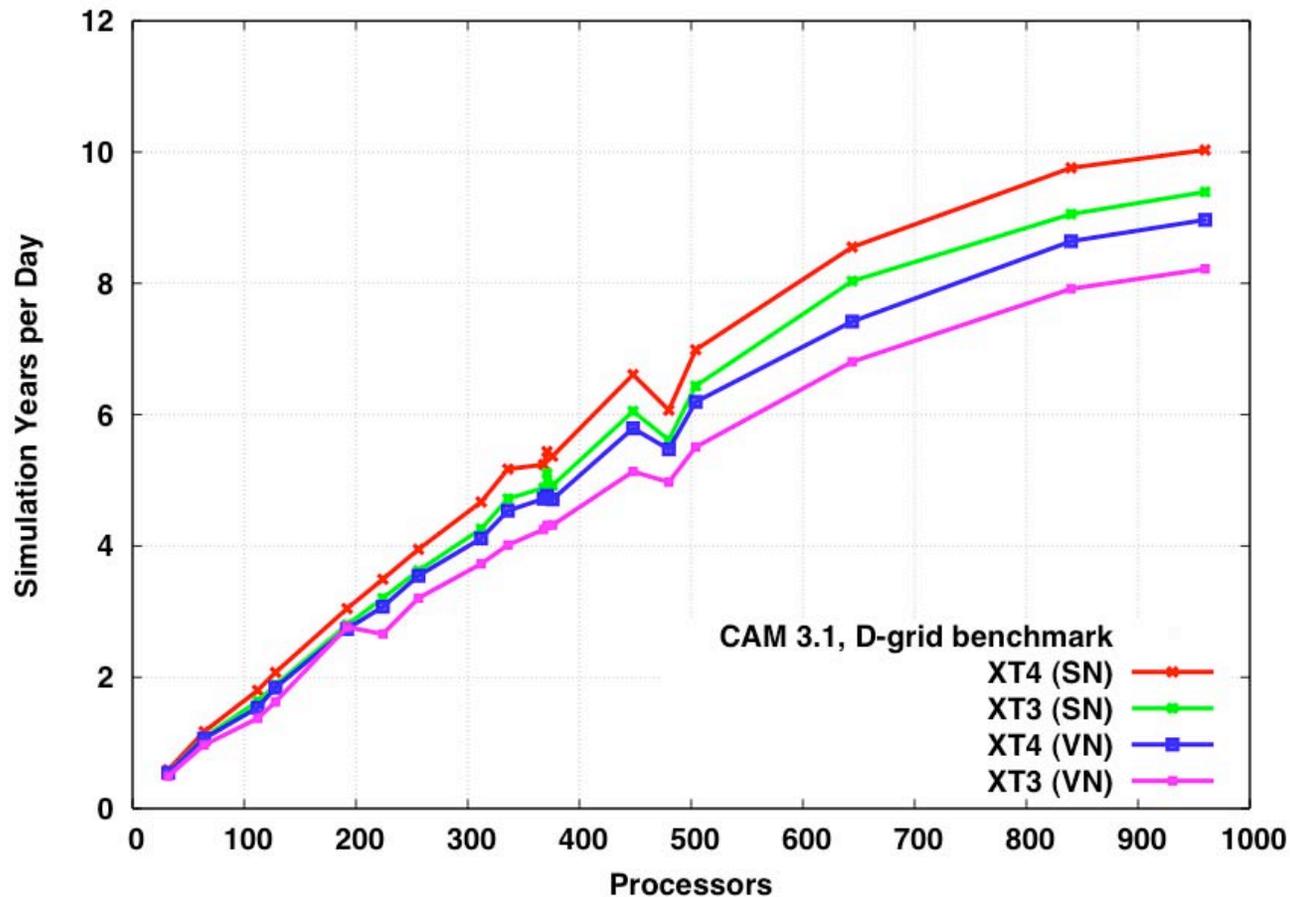


Dynamics performance for 256 and 512 processors (both SN and VN modes) is identical to SN mode performance at 128 processors. The cost of the dynamics-physics coupling decreases out to 512 processors. However, when using the MPI collective implementation, the cost at 512 processors is 50% larger than at 256 processors. XT3 results are qualitatively the same as the XT4 results.

Finite Volume Benchmark

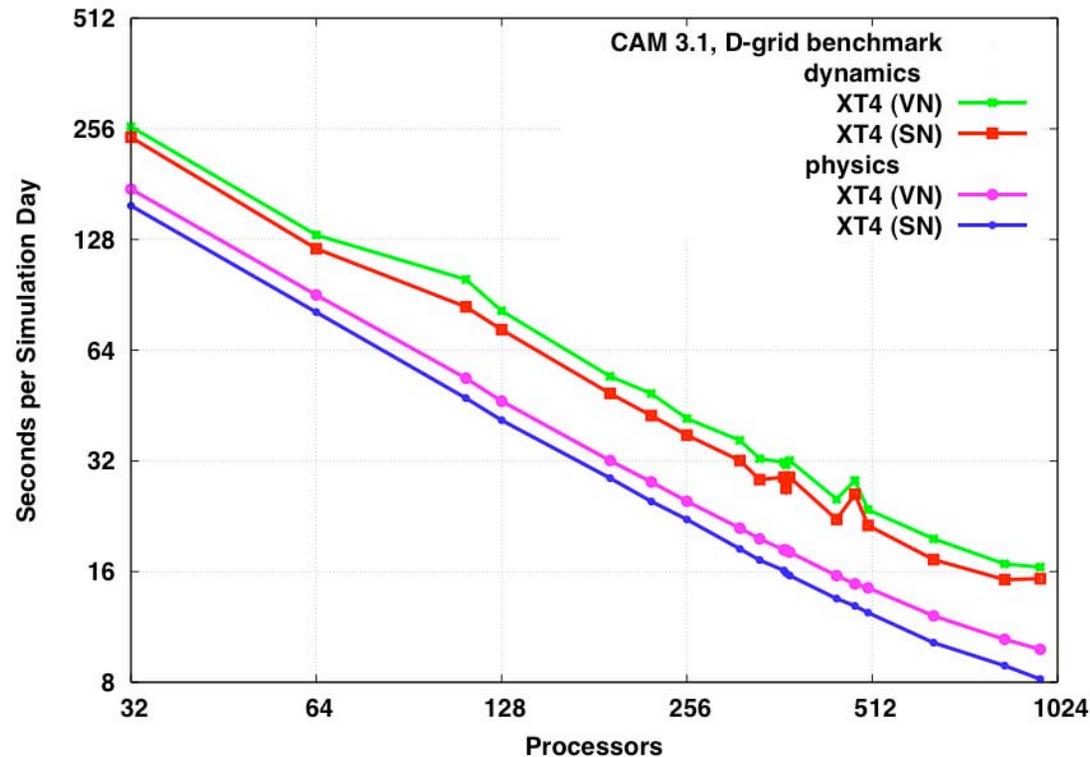
- D-grid: 361x576 latitude-longitude horizontal grid with 26 vertical levels
- Dynamics supports 2D latitude-longitude decomposition in one phase and 2D latitude-vertical in another phase, requiring two remaps per timestep. At least 3 latitudes and 3 vertical levels are required in the decomposition, limiting maximum number of MPI processes to 960.
- Physics supports arbitrary 2D horizontal decomposition, but dynamics is twice as expensive as physics. Using more processors in the physics than in the dynamics would have limited impact on performance scalability.
- Compiled with `-fast`; run with `-small_pages`, SMP-style process placement and physics load balancing.

Finite Volume Performance



Performance scales out to the limit of MPI parallelism. The XT4 is faster than the XT3. SN mode is faster than VN mode for the same processor count. XT3 performance is qualitatively the same as XT4 performance.

Finite Volume Phase Analysis



Physics performance is continuing to scale, but the dynamics performance may have reached a lower bound. SN mode is faster than VN mode, and the difference is greatest in the physics. The physics phase includes the dynamics-physics coupling (13% of the physics cost in SN mode; 17% of the cost in VN mode for the 960 processors). Results for the XT3 are similar.

CAM Benchmarks Summary

- 1) Lack of OpenMP support limits scalability when running in VN mode.
- 2) Supporting different numbers of active MPI processors in different phases is effective in increasing scalability in the T85L26 benchmark.
- 3) Using SN mode in phases using smaller numbers of processors and using “optimal” point-to-point implementations of collectives improves performance further.
- 4) CAM performance results agree with kernel results:
 - 1) XT3 and XT4 performance are qualitatively similar;
 - 2) the XT4 is faster than the XT3;
 - 3) SN mode is faster than VN mode for the same number of processors.

Conclusions

The performance characteristics of the Cray XT3 and XT4 architectures are qualitatively identical, with the same optimization strategies appropriate for both systems. For more details see the paper.

More Caveats

- Quantitative results presented here are application-specific and will change as the system software changes, especially with regards to the efficacy (or lack thereof) of higher levels of compiler optimizations and the impact of MPI environment variables.
- In other benchmarks MPI_Allreduce performance was sensitive to the process placement option. Defining MPI_COLL_OPTION removed this sensitivity, achieving the best observed performance for all (examined) placement options.
- In another benchmark, using PETSc, SMP-style placement degraded performance compared to the default wrap placement.