Performance of a Direct Numerical Simulation Solver for Combustion on the Cray XT3/4

Ramanan Sankaran and Mark R. Fahey
National Center for Computational Sciences
Oak Ridge National Laboratory

Jacqueline H. Chen
Sandia National Labs, Livermore, CA

Collaborators:
Evatt R. Hawkes, Chun S. Yoo and David O. Lignell

Presented at CUG 07, Seattle

OAK RIDGE National Laboratory

NATIONAL CENTER FOR COMPUTATIONAL SCIENCES

Acknowledgements

- Research sponsored by the Mathematical, Information, and Computational Sciences Division, Office of Advanced Scientific Computing Research, U.S. Department of Energy, under Contract No. DE-AC05-00OR22725 with UT-Battelle, LLC.

- The work at SNL was supported by the Division of Chemical Sciences, Geosciences and Biosciences, the Office of Basic Energy Sciences (BES), the U.S. Department of Energy (DOE) and also by the U.S. DOE, BES, SciDAC Computational Chemistry program.

S3D at NCCS

- S3D is a flow solver for performing direct numerical simulation (DNS) of turbulent combustion
  - S3D is a state-of-the-art code developed at CRF/Sandia
  - Has been ported and scales well on most Office of Science platforms
- Project title: "High-fidelity numerical simulations of turbulent combustion - fundamental science towards predictive models"
  - PI: Jacqueline H. Chen (SNL)
  - Past awards: '05 INCITE (NERSC), '05 and '06 LCF (NCCS)
  - 2007 INCITE award - 6M hours on XT3/4 at NCCS
  - A tier1 pioneering application for the 250TF system

Direct Numerical Simulation (DNS)

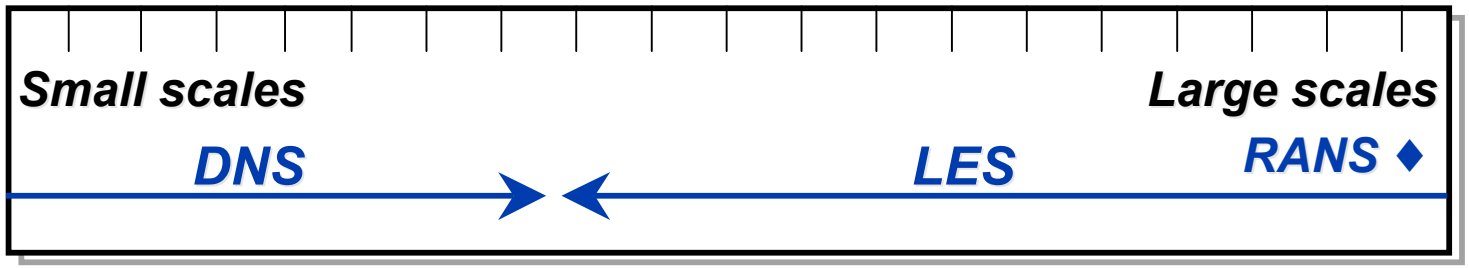
National Center for Computational Sciences

Turbulent Combustion is a Grand Challenge

- Turbulent Combustion involves coupled phenomena at a wide range of scales.
  - Combustor size ~ 1m
  - Flame thickness 10~100μ
- $O(10^4)$ continuum scales.

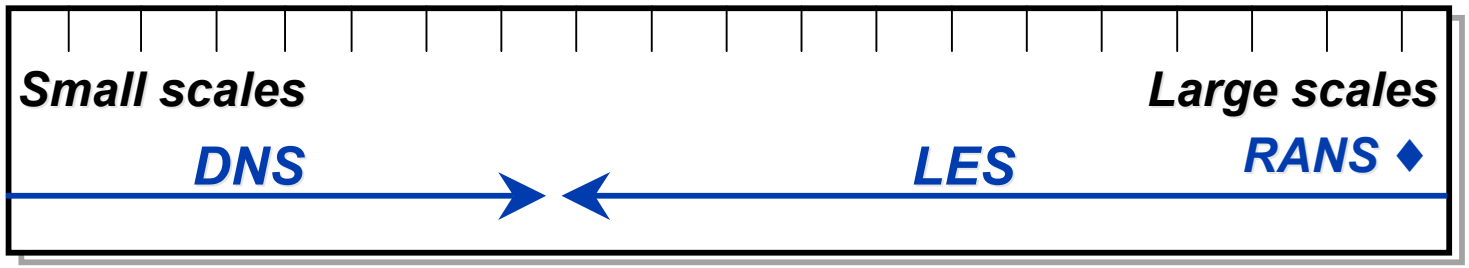
DNS Approach and Role

- Fully resolve all continuum scales without using sub-grid models
- Only a limited range of scales is computationally feasible.
  - Terascale computing = DNS with $O(10^3)$ scales for cold flow.
- DNS is currently limited to small domains. Device-scale simulations are out of reach.



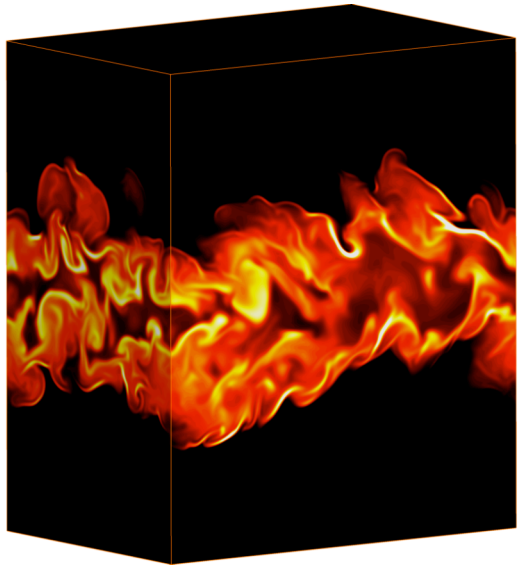
Example of Feasible Domain Size

- Grid spacing dictated by small scales of turbulence and chemistry
  - 15 ~ 30 microns necessary to fully resolve the reaction layer for typical flames
- 1 million grid points = 100 x 100 x 100
  - Domain size = 1.5 ~ 3mm
- 1 billion grid points = 1000 x 1000 x 1000
  - Domain size = 15 ~ 30mm
- Larger domain = wider range of scales
  - Closer to reality

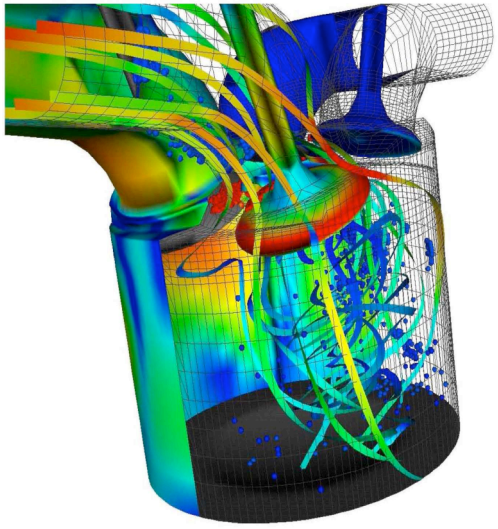


Role of Direct Numerical Simulation

- DNS is a tool for fundamental studies of the micro-physics of turbulent reacting flows
  - Full access to time resolved fields
  - Physical insight into chemistry turbulence interactions



- A tool for the development and validation of reduced model descriptions used in macro-scale simulations of engineering-level systems



DNS ↔ *Physical Models* ↔ *Engineering CFD codes (RANS, LES)*

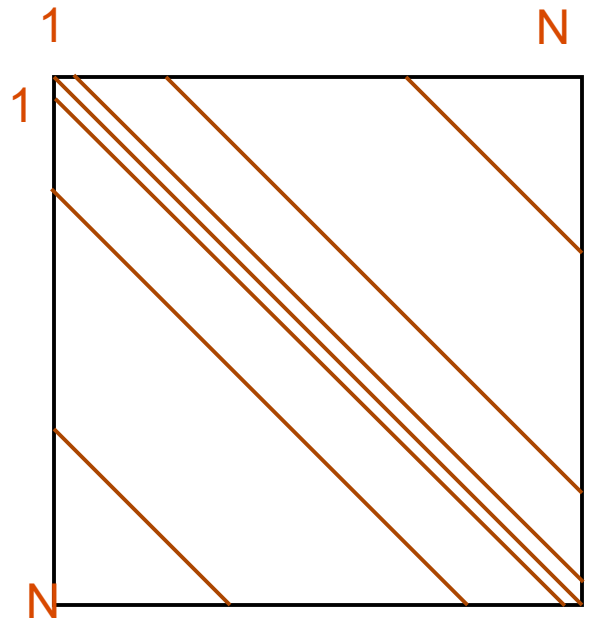
S3D - DNS Solver

S3D is a state-of-the-art DNS code developed at CRF/Sandia with 13 years of BES sponsorship.

- Solves compressible reacting Navier-Stokes equations.
- High fidelity numerical methods.
  - 8th order finite-difference
  - 4th order explicit RK integrator
- Hierarchy of molecular transport models
- Detailed chemistry
- Multi-physics (sprays, radiation and soot)
  - From SciDAC-TSTC (Terascale Simulation of Turbulent Combustion)
- Fortran90 and MPI
- Highly scalable and portable

Parallelism

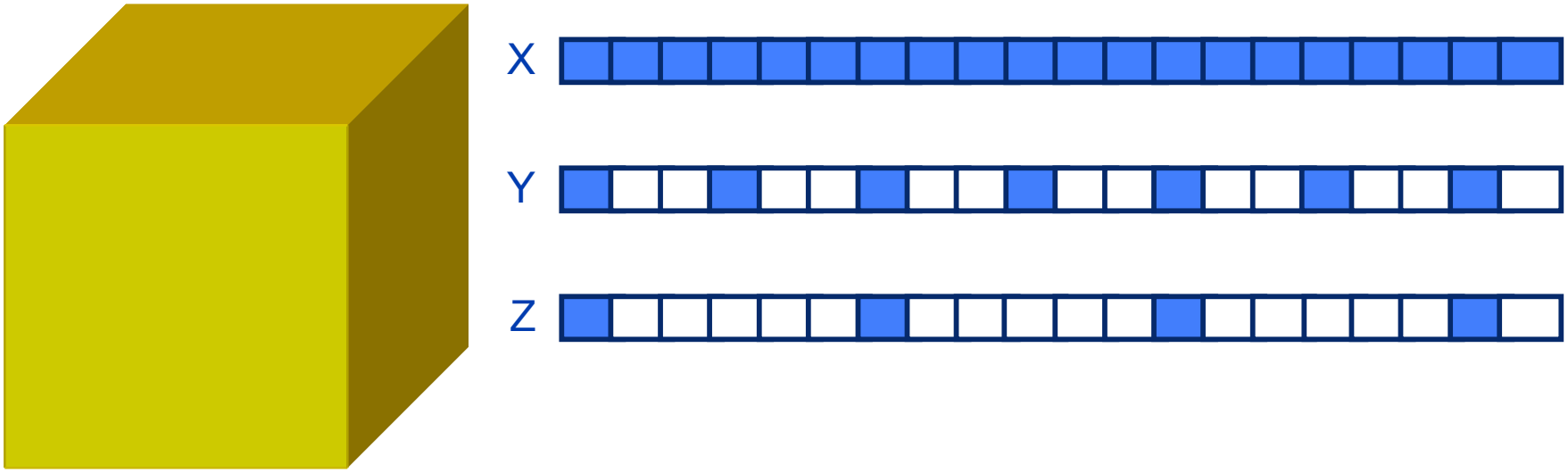
- Parallelism is achieved through 3D domain decomposition.
  - Each MPI process is in charge of a piece of the 3D domain.
- All MPI processes have the same number of grid points and the same computational load
- Inter-processor communication is only between nearest neighbors in 3D topology
  - Large message sizes. Non-blocking sends and receives
- All-to-all communications are only required for monitoring and synchronization ahead of I/O



$$\frac{\text{Communication}}{\text{Computation}} = \frac{kN^2}{kN^3} = \mathrm{O}\left(\frac{1}{\mathrm{N}}\right)$$

Non-contiguous array communication

- Boundary planes of a 3D array needs to be communicated to neighbors



- In 2 out of 3 directions, communication involves a non-contiguous array with a constant stride

What can be done with this code?

- Create a MPI data type to represent the strided data.
  - Done only once during initialization.

```
call MPI_Type_vector(   nz,    nx*4, nx*ny, MPI_REAL8, yrows_type, ierr)
call MPI_Type_commit(yrows_type,ierr)
```

- Use it for MPI communication

```
call MPI_ISend(f(1,1,1),1,yrows_type,lnbr(3),1,gcomm,req(3),ierr)
```

- Nothing is wrong. But the following is much faster

```
pos_fs(:,:,:) = f(:,1:4,:)
call MPI_ISend (pos_fs(1,1,1), mx*mz*4, MPI_REAL8, lnbr(3), 1, gcomm,
    req(1), ierr)
```

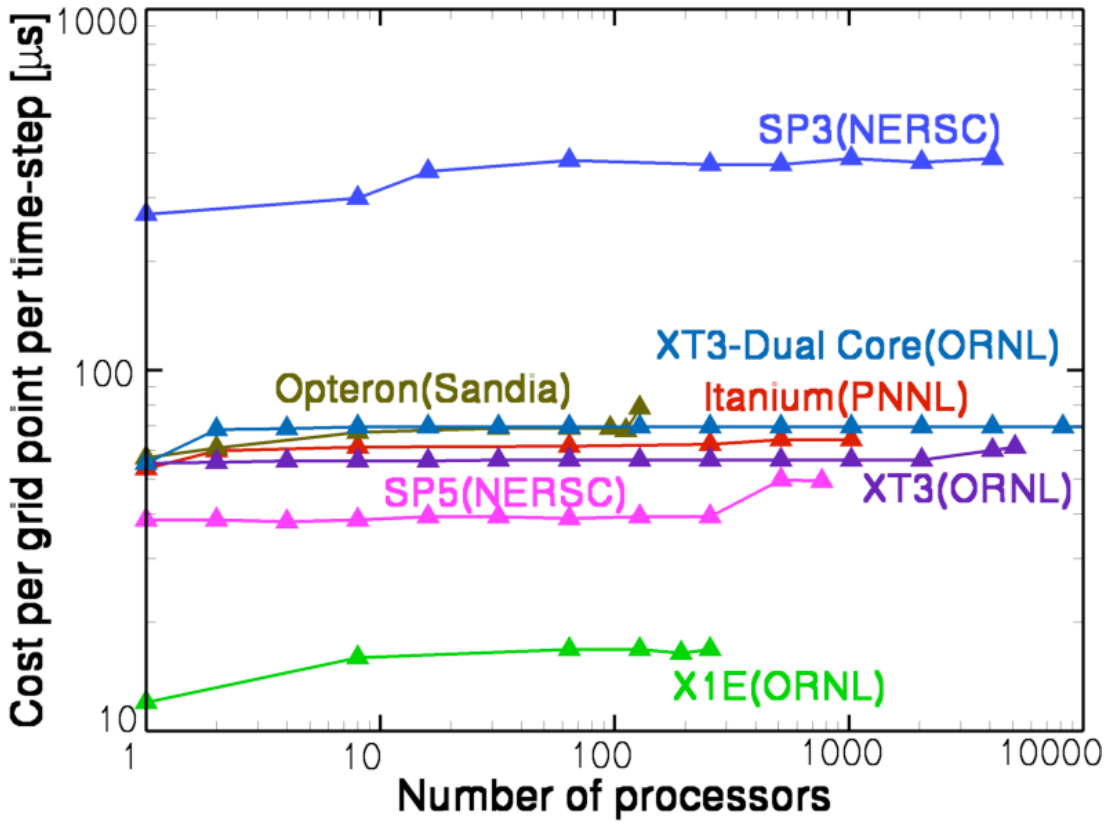
Demonstrated Parallel Performance

- S3D has been used to perform several hero simulations in the past couple of years
  - Enabled by large INCITE awards
  - Continuous effort in porting and optimizing code on evolving architectures

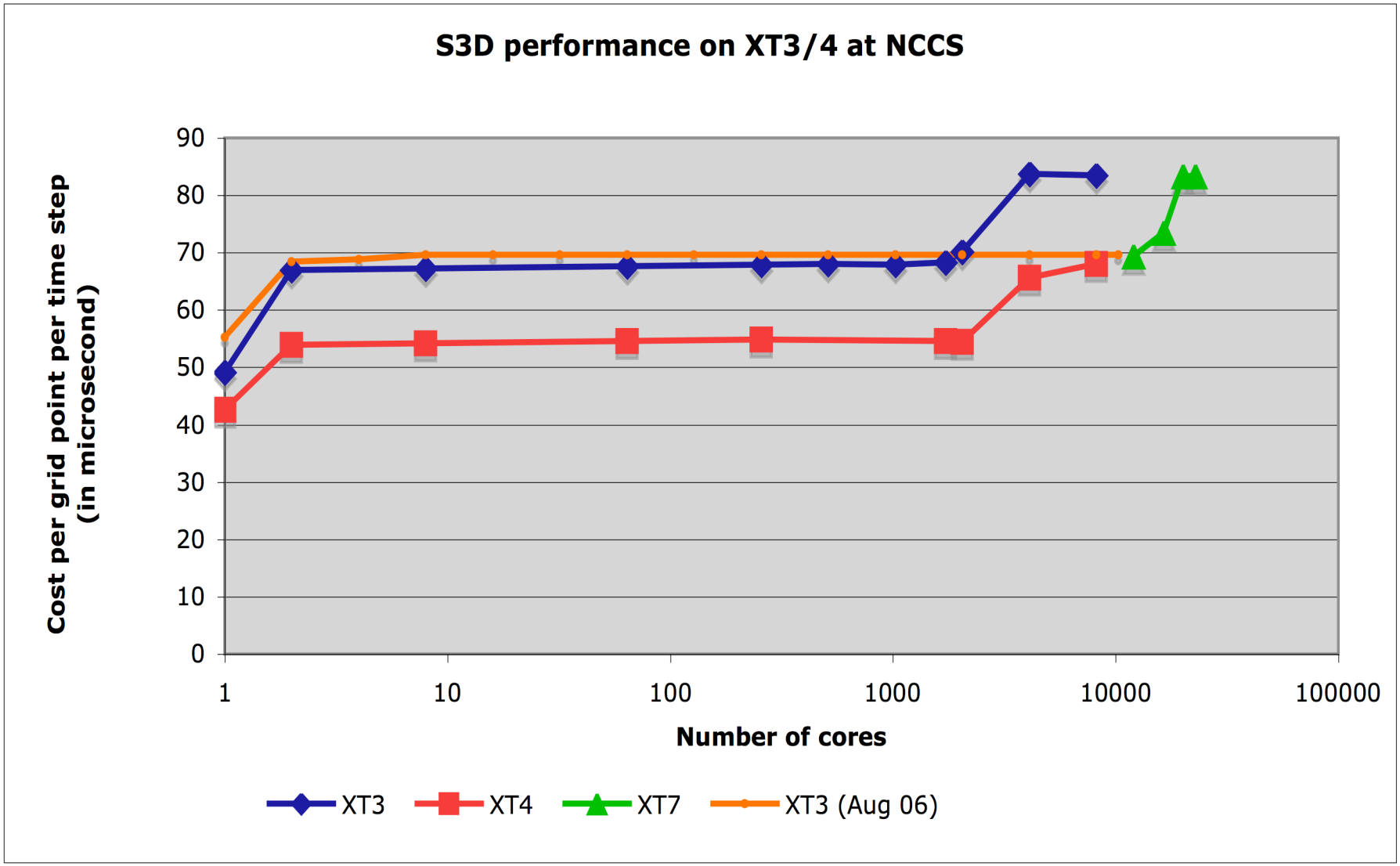
Details on recent S3D production simulations		
Oct 05	150M grid points, 16 variables	Itanium cluster at PNNL
Nov 05	350M grid points, 16 variables	600 processors on IBM SP5 at NERSC
Nov 05	500M grid points, 16 variables	512 MSPs on X1E at NCCS
Dec 05	52M grid points, 18 variables	512 MSPs on X1E at NCCS
May 06	88M grid points, 18 variables	4800 processors on XT3 at NCCS
Sep 06	194M grid points, 18 variables	7200 cores on dual-core XT3 at NCCS
Dec 06	1B grid points, 14 variables	9000 cores on dual-core XT3 at NCCS
May 07	350M grid points, 24 variables	Planned run of 12K cores on dual-core XT3/4 at NCCS

S3D's Parallel Scaling

- Measure of performance: cost of execution per grid-point per time-step
- Weak scaling test with 50^3 grid points per MPI thread

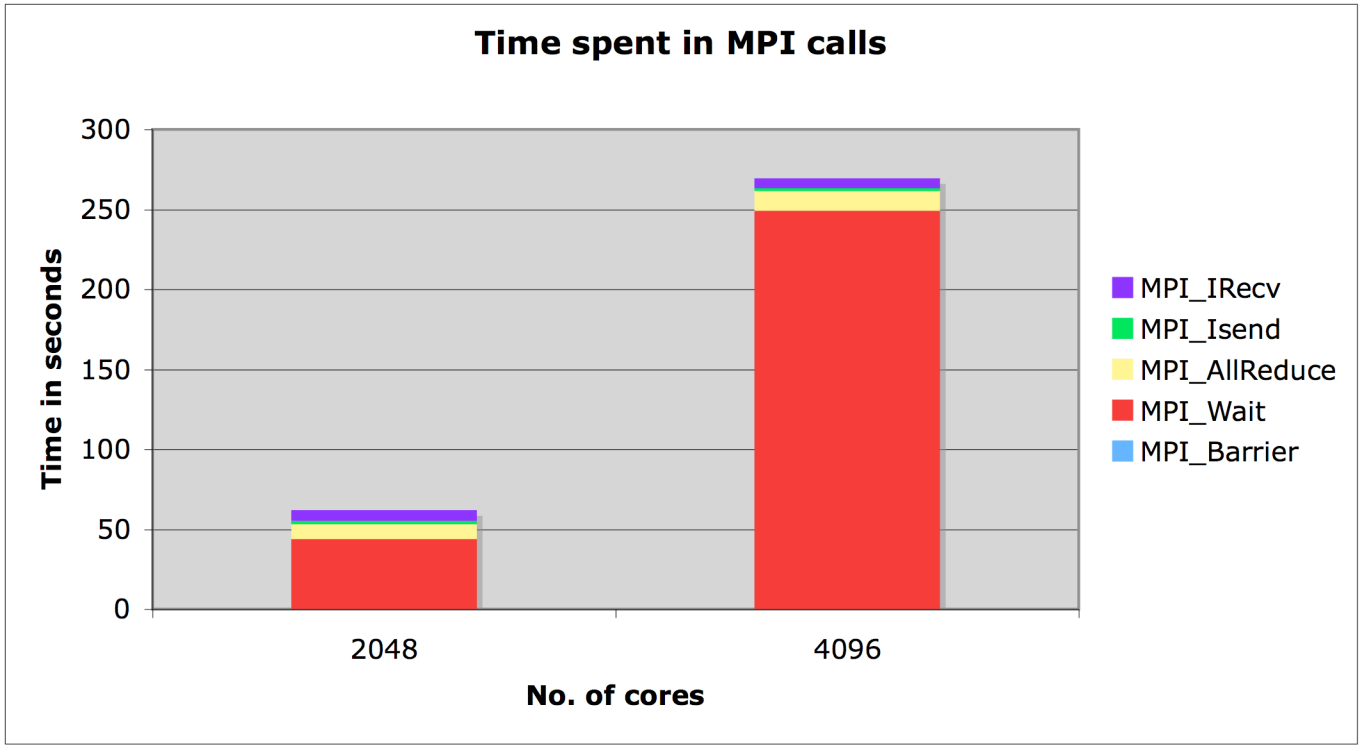


Performance on XT3+4 at NCCS



MPI communication times from fpmpi

- Benchmark run took 877s on 2048 XT3 cores and 1047s on 4096 XT3 cores (similarly on XT4)
  - Weak scaling test with 50^3 grid point per mpi thread
- Most of the increase is from MPI_Wait on non-blocking sends and receives



Representative derivative code

```
if(lnbr(1)>=0) then
 ! get ghost cells from neighbor on (-x) side
   call MPI_IRecv(neg_f,(my*mz*iorder/2),MPI_REAL8,lnbr(1),1,gcomm,req(1),ierr)
 ! send ghost cells to neighbor on (-x) side
   pos_fs(:,:,:) = f(1:iorder/2,:,:)
   call MPI_ISend(pos_fs(1,1,1),(my*mz*iorder/2),MPI_REAL8,lnbr(1),2,gcomm,req(2),ierr)
endif
if(lnbr(2)>=0) then
 ! get ghost cells from neighbor on (+x) side
   call MPI_IRecv(pos_f,(my*mz*iorder/2),MPI_REAL8,lnbr(2),2,gcomm,req(3),ierr)
 ! send ghost cells to neighbor on (+x) side
   nm = mx + 1 - iorder/2;       neg_fs(:,:,:) = f(nm:mx,:,:)
   call MPI_ISend(neg_fs(1,1,1),(my*mz*iorder/2),MPI_REAL8,lnbr(2),1,gcomm,req(4),ierr)
endif

[…computation…]

      if( lnbr(1) >= 0 )then
        call MPI_Wait(req(1),stat(:,1),ierr)
      endif

[…more computation…]

     if( lnbr(2) >= 0 ) then
        call MPI_Wait(req(3),stat(:,3),ierr)
      endif

[….more der calcs and waits…]
```

I/O Strategy

- Restart data is also the analysis data
- Data written to disk ~ once per hour
- Each MPI thread writes to a separate file
- Unformatted fortran binary I/O
- Lustre stripe width set to 1. No striping
- At large job sizes, simultaneous file opens bog down the file-system
  - Possible solutions
    - Block the I/O to limit the maximum number of simultaneous opens. (Implemented and ~1000 works fine).
    - MPI I/O (future work)

Multi-core performance

Fixed Problem size	MPI mode	XT3		XT4	
		wall time	cost	wall time	cost
		seconds	μseconds per gridpoint per timestep	seconds	μseconds per gridpoint per timestep
50x50x50	yod -np 1 -SN	617	49	532	43
100x50x50	yod -np 1 -SN	1206	48	1041	42
100x50x50	yod -np 2 -SN	614	49	532	43
100x50x50	yod -np 2 -VN	838	67	674	54

- Cost of executing the same (constant size per socket) problem is higher on a dual-core processor than a single-core processor
  - craypat shows sections of code operating on large 5D arrays went up in cost
  - In SN mode, a second core is idle though
- The simulation does complete faster wrt wall-clock time when both cores of the dual-core processor are utilized

* Very relevant test sizes - similar to production runs listed earlier

Partnerships in S3D Improvement

- **Performance improvement**
  - **Performance engineering research institute (PERI). David Bailey (LBNL), John Mellor Crummey (Rice U.), Sameer Shende (U. Oregon) and Bronis de Supinski (LLNL)**
- **Efficient parallel I/O**
  - **SDM center. Wei Keng Liao and Alok Choudhary (Northwestern U.)**

Summary

- S3D is a flow solver for performing direct numerical simulation (DNS) of turbulent combustion
  - DNS is a tool for fundamental studies of the micro-physics of turbulent reacting flows
- Demonstrated parallel scaling
  - Nearest neighbor communication
  - Parallel I/O with ability to use a subset of processes
- S3D has been used to perform several hero simulations in the past couple of years
- S3D scaled perfectly on XT3 (before combination)
- Now, on XT3, XT4, or XT3+4
  - scales perfectly to < 4K cores
  - MPI_Wait blows up for 4K cores or more
    - We don't know why
    - Need to understand why before the next large run is started