# *Simulating Laser-Plasma Interaction in Experiments at the National Ignition Facility on a Cray XE-6*

Cray Users Group 2012

May 2, 2012

Steve Langer

Lawrence Livermore National Laboratory

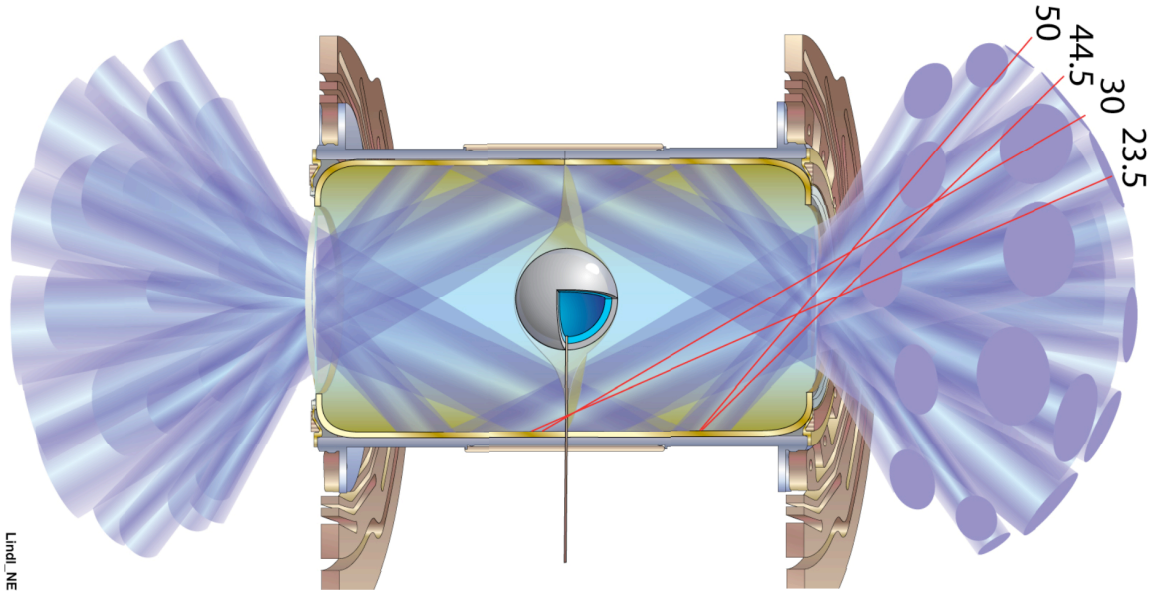
LLNL-PRES-547711

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC

# pF3D has run several large simulations on Cielo and is undergoing preparation for many-core machines

- pF3D simulates the interaction between an intense laser beam and the ionized gas through which it travels.
- A recent simulation on Cielo ran for several months (7 CPU millenia) on 32,768 cores and had over 220 billion zones.
- pF3D performs well on Cielo.
- Custom domain to interconnect location software is being developed to better exploit toroidal interconnects.
- OpenMP parallelism is being added to pF3D to better exploit many core chips.
- More efficient checkpointing schemes are being added to pF3D to improve our resiliency.

# pF3D simulates interactions between laser beams and the plasma in a NIF hohlraum




- The National Ignition Facility (NIF) is NNSA's premier high-energy density physics experimental facility.
- The NIF laser is the most energetic pulsed laser in the world.
- NIF experiments are expensive and there are a limited number per year.
- Accurate simulations can have a large positive impact on the productivity of NIF.

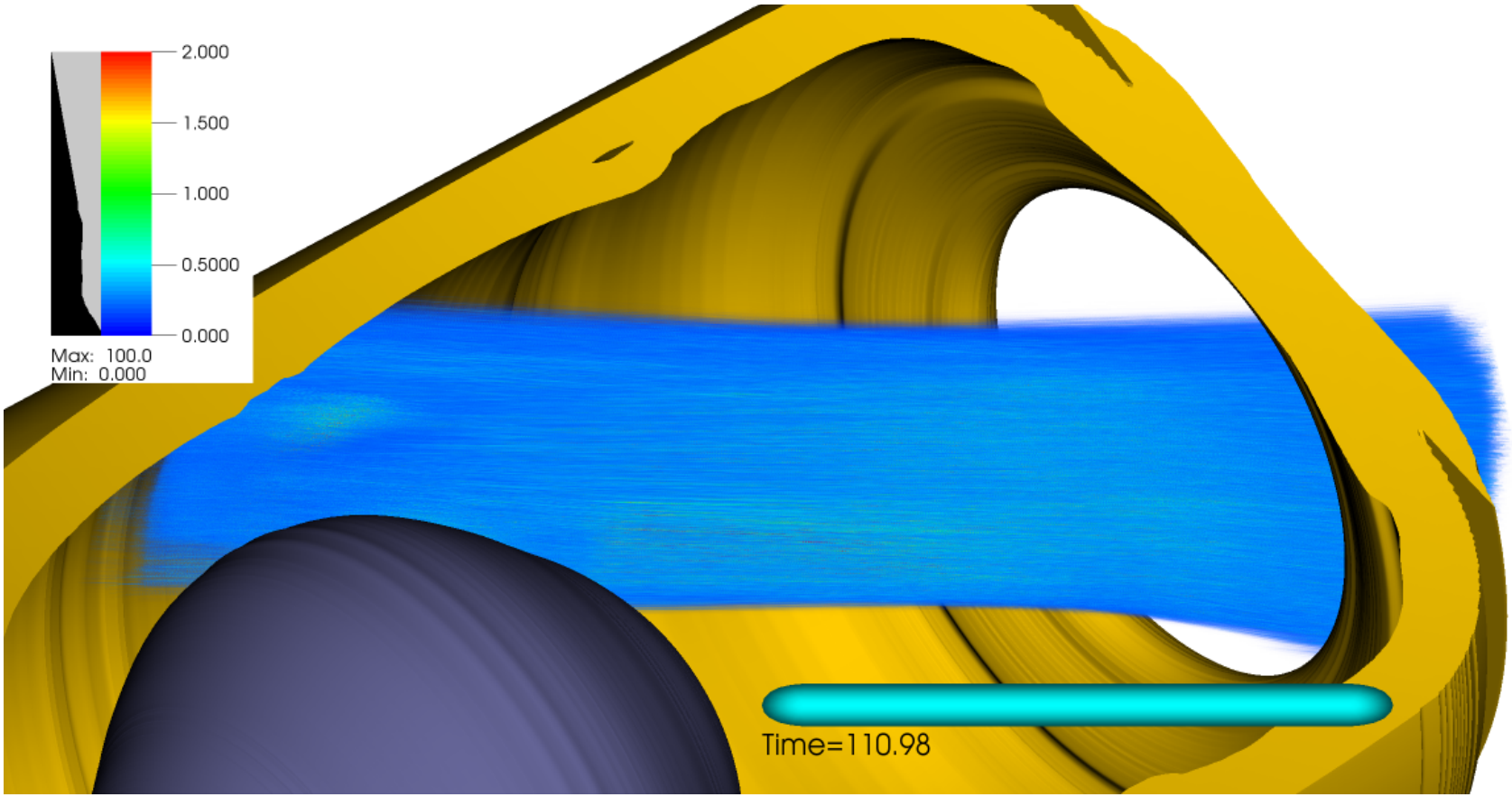
- The laser beams enter through holes in the ends of the hohlraum.
- There are 192 beams in 48 beam quads.
- The plasma in the hohlraum is heated to a few keV.
- The laser beams deposit their energy near the wall of the hohlraum.

- The walls are heated to roughly 300 eV and emit x-rays.
- The x-rays ablate material from the capsule and drive an implosion.
- The laser light can interact with fluctuations in the ion or electron density to scatter some light back out of the hohlraum.
- We simulated beams in the 23.5 and 30 degree cones.

# pF3D is a multi-physics code used to simulate laser-plasma interactions

- pF3D includes multi-material Eulerian hydrodynamics on a regular Cartesian grid
- Wave solvers are used to update the laser light, Stimulated Brillioun Backscatter, Stimulated Raman Backscatter.
- Each light wave has two polarizations.
- pF3D also solves for electron plasma wave and ion-acoustic wave amplitudes.
- Wave interactions are solved in the paraxial approximation using 2D FFTs.
- There are 25 light cycles per hydro step, so the light computation dominates the run time.
- Flexible input decks, graphics, and I/O are provided by the yorick interpreted language (yorick.sf.net).
- pF3D is written in C and uses domain decomposition and MPI message passing to achieve parallelism.

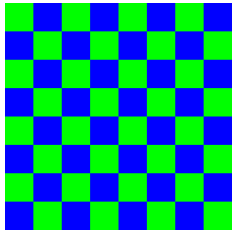
# The laser beam crosses several mm of plasma between the entrance hole and the hohlraum wall



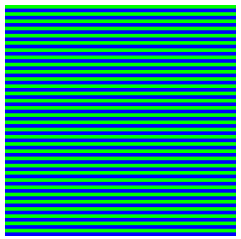
# Wave coupling and propagation require distributed 2D FFTs

- pF3D makes the paraxial approximation (i.e. the waves move nearly in the z-direction).
- The coupling between the different waves is solved using parallel 2D FFTs in planes of constant z.
- An xy-plane is split into NxM domains.
- MPI messages transpose from a "checkerboard" decomposition to rows to columns and back to a "checkerboard".
- Single processor 1D FFTs are performed while in the row and column decompositions.
- pF3D performs many 2D FFTs so high message passing rates are needed to achieve good performance.

normal decomposition ("checkerboard")

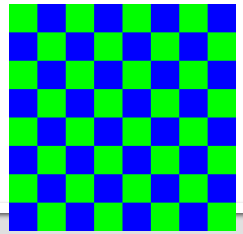


transpose to rows perform 1D FFT



transpose to cols perform 1D FFT

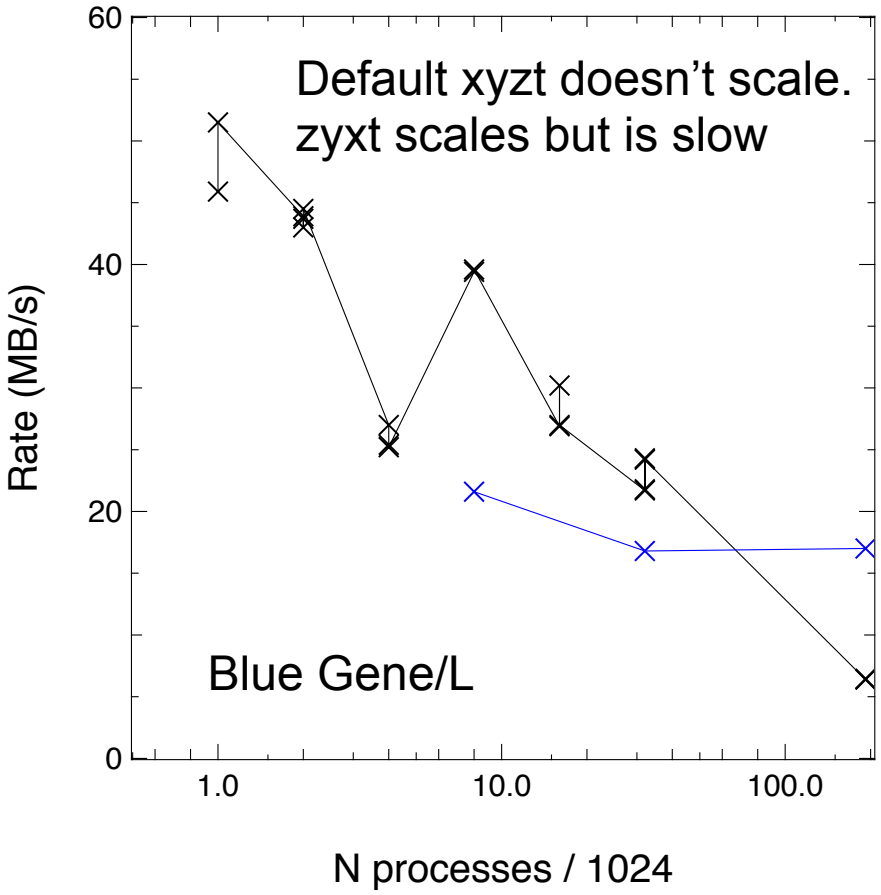
transpose to checkerboard

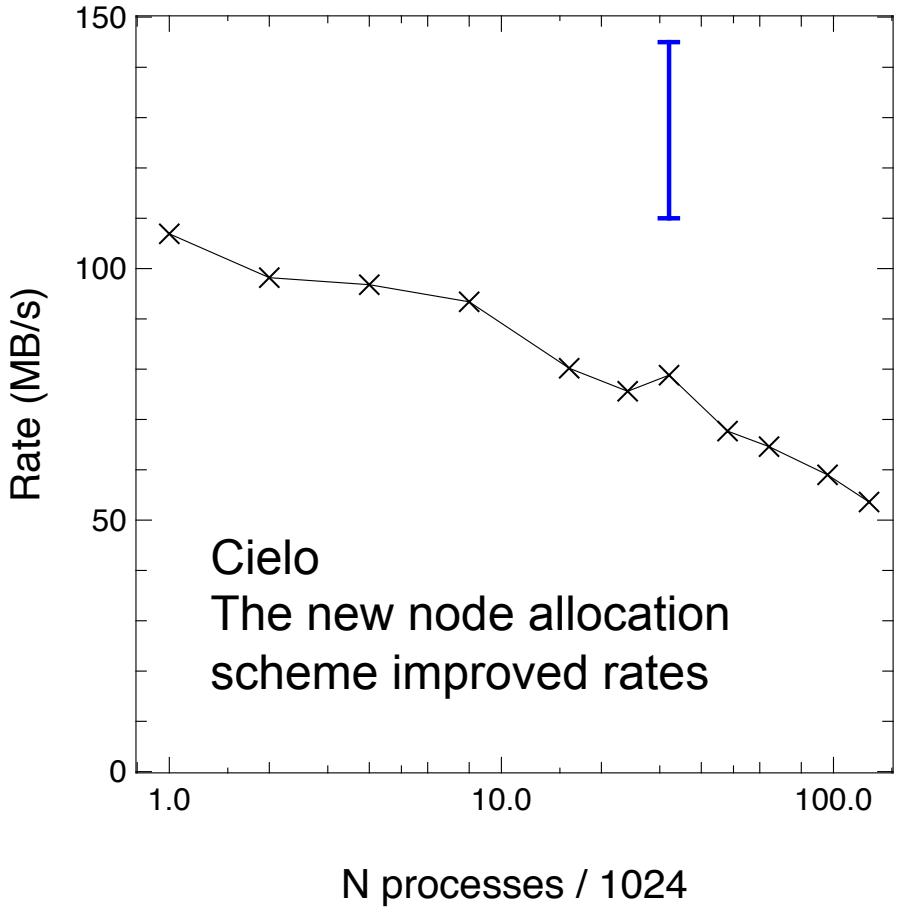


# The mapping from spatial domains to location on the interconnect matters on Cray XE-6 and Blue Gene systems

- The maximum hop count on fat tree interconnects grows slowly with node count.
- Messages passed on large totoidal interconnects may need to make many hops. Large hop counts tend to reduce message rates.
- Contention for links is a problem for long messages and latency is a problem for short messages.
- pF3D processes sharing an xy-slab communicate heavily with each other and less heavily with other processes.
- Custom schemes for mapping MPI domains to torus locations can improve message passing performance.
- Mapping schemes need to be aware of the groups of processes that communicate heavily with each other.

# Scalable pF3D message passing schemes are available on both BlueGene systems and Cielo



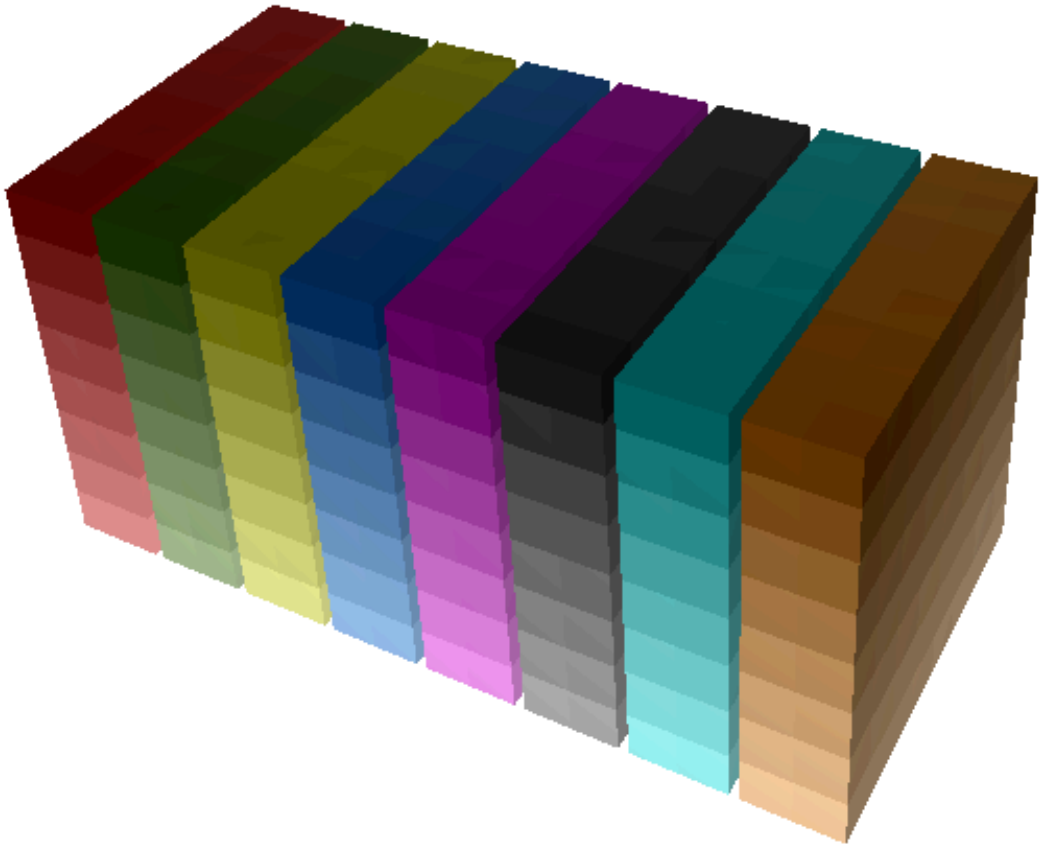
# Cray XE-6 and IBM Blue Gene systems pose different challenges in achieving high message rates

- pF3D is a bulk synchronous application. Application performance is controlled by the slowest communicator.
- On Cielo, the default shape for a 16 process y-communicator was 1x2x8 before Sep. 2011 and 2x2x4 starting in Sep. 2011.
- Some communicators have a longer extent due to skipping nodes reserved for I/O, nodes that are off line, and nodes in use by other jobs. These "stretched" communicators control application performance on Cielo.
- BG/L and BG/P systems assign a private torus to each application. Mappings can easily be set up where all communicators have the same shape and hence the same speed.
- Key performance issues on BG/L and BG/P systems are utilizing all three pairs of links and keeping the hop count relatively low.

## Rubik simplifies the generation of custom mappings for Blue Gene systems

- Rubik is a custom mapping generator developed at LLNL.
- pF3D users split the computational domain into regular 3D chunks.
- Rubik commands split the interconnect up into regular chunks. The splitting works for the 3D torus of a Blue Gene/P and for the 5D torus of a Blue Gene/Q.
- Rubik maps computational domains to chunks on the torus.
- Rubik can apply “tilting” operations to the domain locations. Tilting can increase the number of links on the torus that are used during a given message passing phase and thereby increases message rates.
- We are investigating ways to extend Rubik to Cray XE systems where processes are assigned an irregularly shaped set of nodes on the underlying torus.

# Standard mappings take an xy-slab to a plane of the interconnect



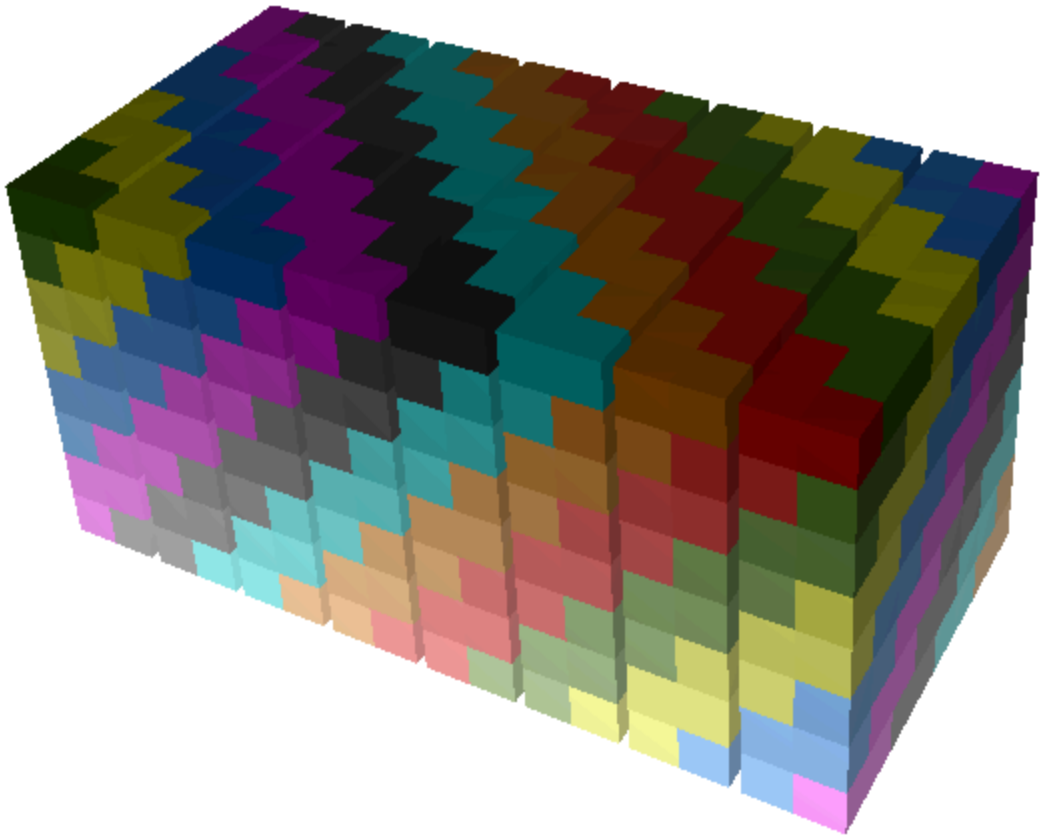
This simulation is running on an 8x8x8 partition of a Blue Gene/P.

xy-slabs are decomposed into 8x8 domains.

Each xy-slab is mapped to an 8x8 plane of the interconnect.

One pair of links is used for x FFT messages and another pair is used for y FFT messages.

# Tilted mappings allow pF3D to use more sets of links on Blue Gene L and P systems



The 8x8x8 mapping has been “tilted” in two directions.

X- and y-communicators become 45 degrees lines on the torus.

x and y FFT messages can use two pairs of links.

More complex mappings enable the use of all links.

# Rubik can generate high performance mappings on a Blue Gene/P

- The message passing rate on an 8x8x8 partition using the default xyzt mapping is 55 MB/s. The fastest mapping generated by Rubik delivers 195 MB/s. This mapping applies tilts in both the y and z directions.
- Message passing rates are lower on large partitions. The default xyzt mapping delivers 8 MB/s on a 48x32x16 partition of LLNL's Dawn BG/P system. A Rubik mapping based on 8x8x8 chunks with two tilts delivers 31 MB/s.
- pF3D can take advantage of the toroidal character of the interconnect only if an xy-slab extends more than half the torus width. Smaller chunks will see the interconnect as a mesh.
- Confining an xy-slab to a localized portion of the torus will leave some links unused, but the lower hop counts may provide a net benefit.

# pF3D needs to exploit systems with a rapidly increasing number of cores per chip

- The Intel Sandy Bridge Xeon has 8 cores with 2 hardware threads per core.
- The IBM BlueGene/Q has 16 cores with 4 hardware threads per core.
- The Intel Knights Ferry many core (MIC) chip has 32 cores and 4 hardware threads per core.
- The Nvidia Fermi board has 448 cores.
- Full performance on some of these chips can only be reached with two or more hardware threads per core.
- Using all hardware threads as MPI processes may not be the best way to exploit these chips.

# pF3D will use OpenMP to exploit large numbers of hardware threads per chip

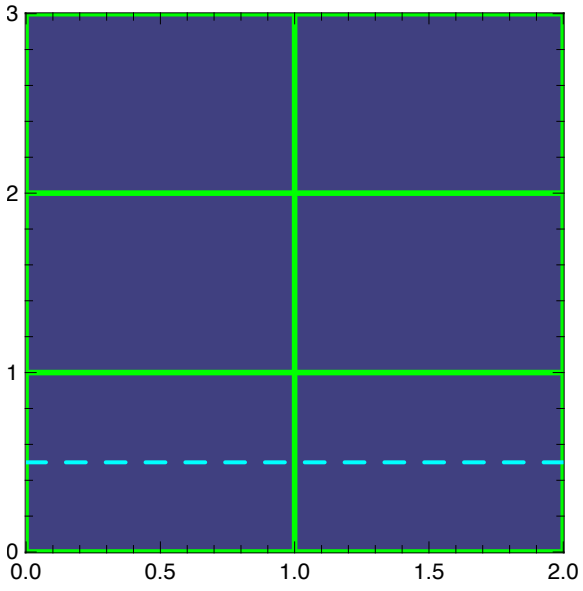
- Message sizes for pF3D decrease as the number of zones per process decrease. Message passing becomes slower if the message size becomes too small.
- OpenMP parallelism will allow pF3D processes to maintain large message sizes even though the memory per hardware thread will drop. The process, not individual threads, will pass the messages.
- OpenMP is fairly simple to add to a code that operates on large 3D arrays.
- OpenACC directives can be added to the OpenMP loops on systems with accelerators.
- OpenMP loops have overhead due to starting threads and splitting up the work at the start of the loop and clearing a barrier at the end of the loop.
- Threads need to have enough zones that the compute time per thread is larger than the thread overhead time.

# The message size drops rapidly as the number of processes in an xy-plane increases

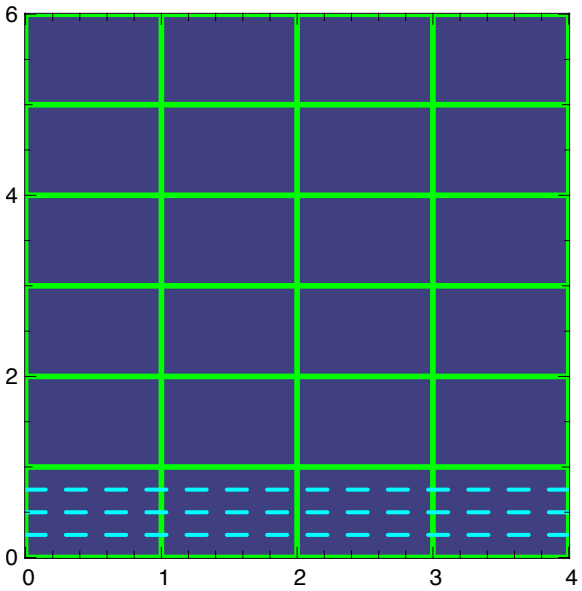
Messages are passed using MPI_Alltoall.

Message sizes are proportional to the number of zones per process and inversely proportional to the number of processes in a group.

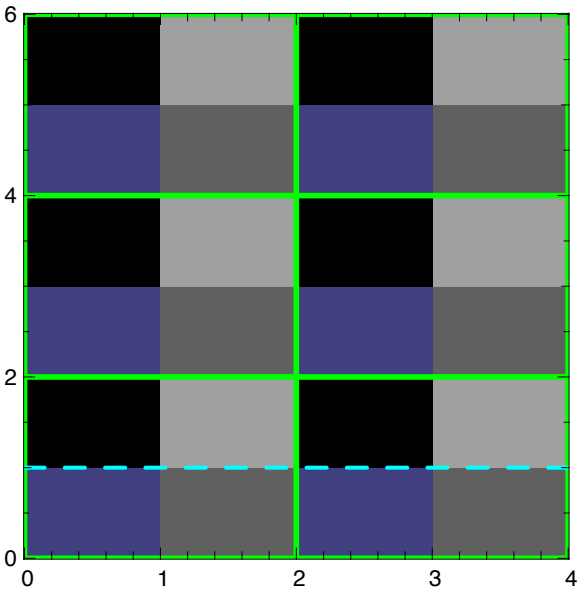
Decomposition for 2 GB per process




Decomposition for 0.5 GB per process



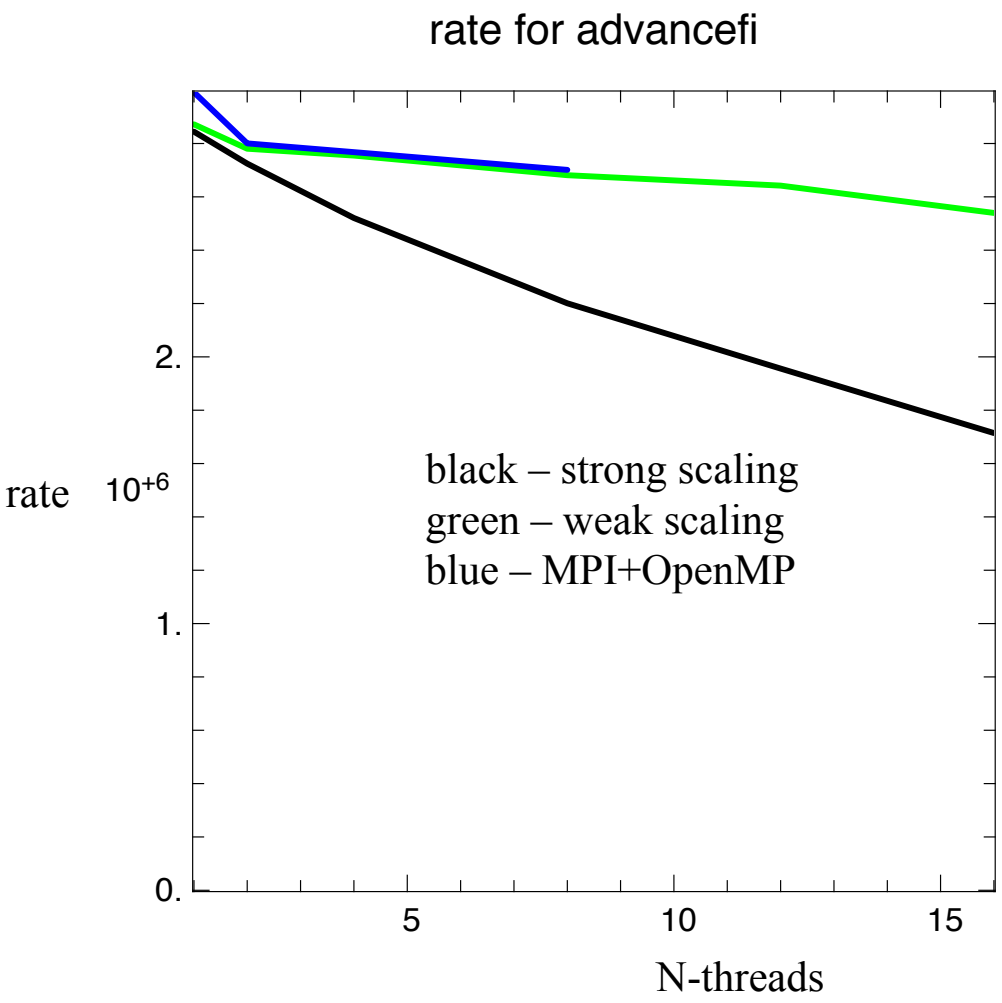

Decomposition for 2 GB and 4 threads per process



# OpenMP tests were run on three recent x86_64 chips

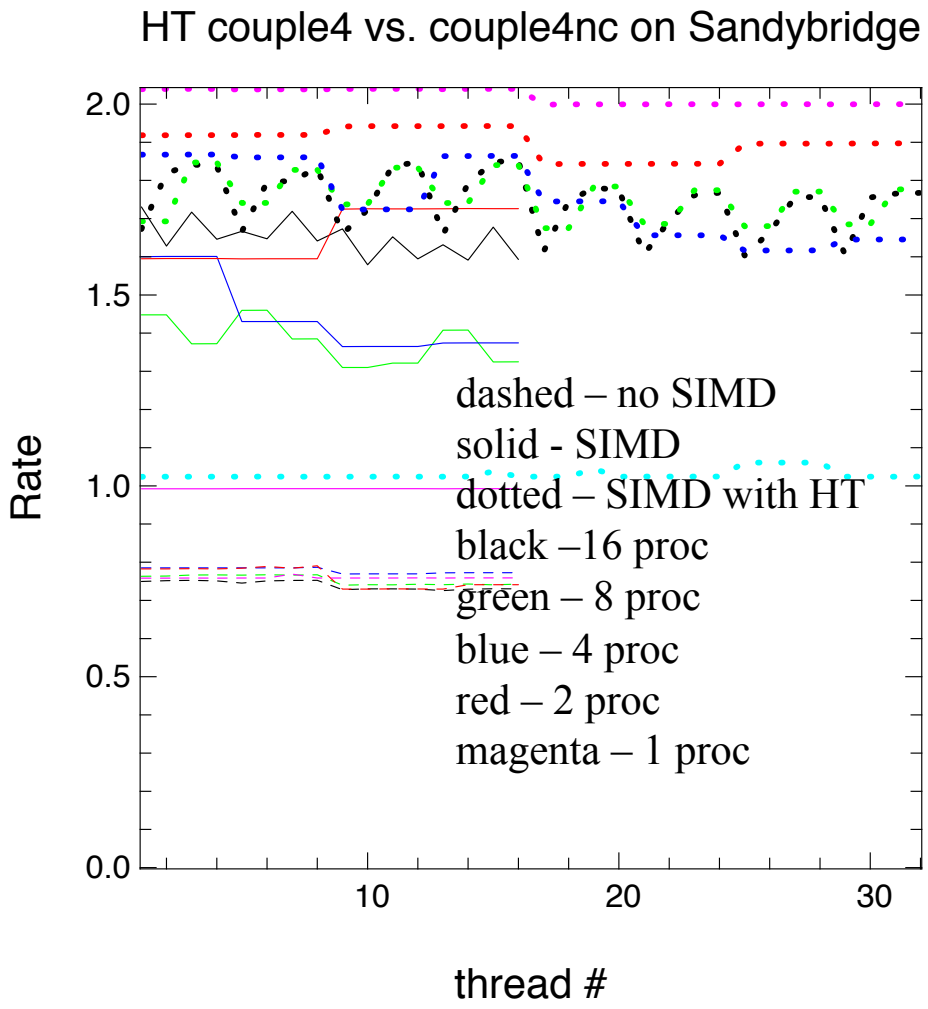
- Intel Sandy Bridge (SB) chips have 8 cores per socket and an AVX 256 bit floating point unit. The SB has two hardware threads per core. The test system had two sockets.
- AMD Magny-Cours (MC) chips in Cielo have 8 cores per socket with 2 NUMA domains per socket and a 128 bit SSE4 floating point. The test system had two sockets.
- The AMD Interlagos uses the Bulldozer architecture (BD) and has 8 floating point units per chip. Each floating point unit is shared between two integer cores. The floating point unit can operate as a 256 bit AVX or a 128 bit SSE4 unit. The test system had two sockets.

# Good OpenMP scaling requires enough work per thread




- Strong scaling – hold the total number of zones constant as the number of threads increases.
- Weak scaling – hold the number of zones per thread constant as the number of threads increases.
- MPI+OpenMP
  - $N_{threads} * N_{procs}$ is held constant.
  - Threads have the same number of zones in all cases.
  - The zone count summed over processes is independent of the number of threads.
- Strong scaling shows a drop off in efficiency for the higher thread counts.
- The rate is nearly constant for weak scaling and the MPI+OpenMP code.

# Thread balance is pretty good for couple4 on a Sandy Bridge node

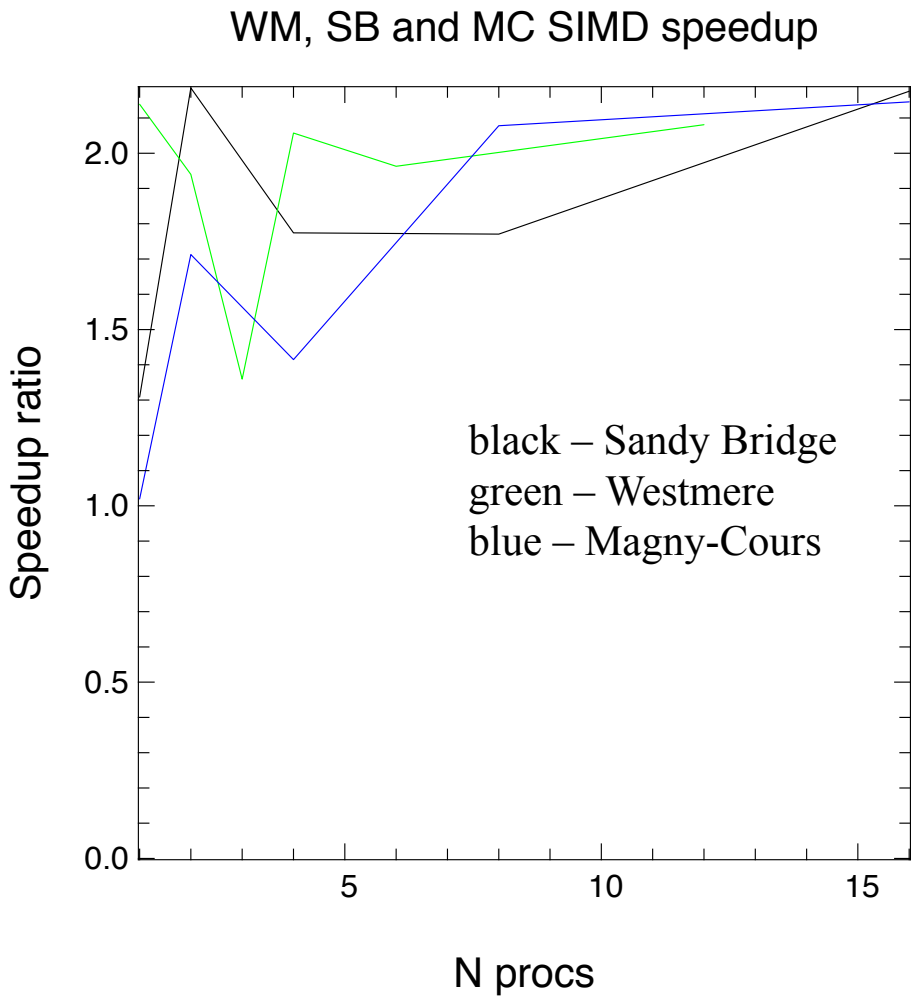



SIMD vectorization and hyperthreading increase the throughput of a core.

Thread-to-thread variability increases when using either SIMD vectorization or hyperthreading.

A single process uses the memory bandwidth of a single socket. Multiple processes use the bandwidth of both sockets. Multi-threaded processes are faster for the SIMD version.

# SIMD vectorization increases the throughput of couple4 by ~2X

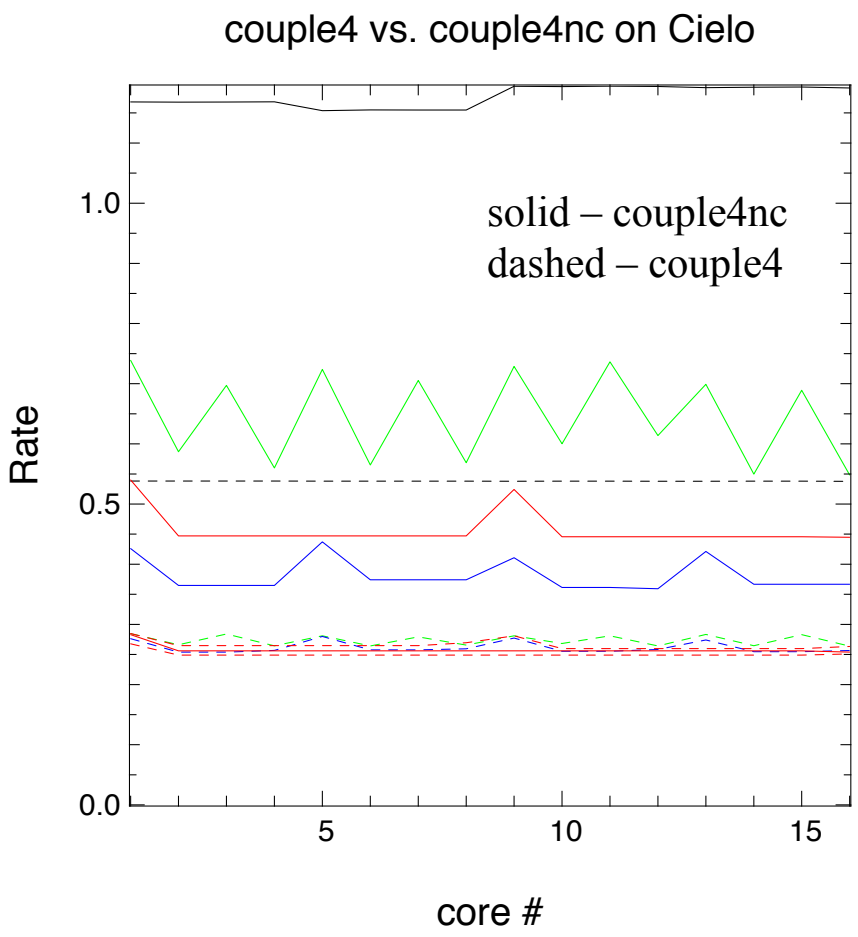



SIMD vectorization increases the throughput of a core.

Speedups for a single process are lower due to memory bandwidth limitations.

The speedup is similar for Westmere and Sandy Bridge in spite of the wider SIMD registers of the Sandy Bridge.

# SIMD speedup rates for couple4 are not yet available on Magny-Cours chips

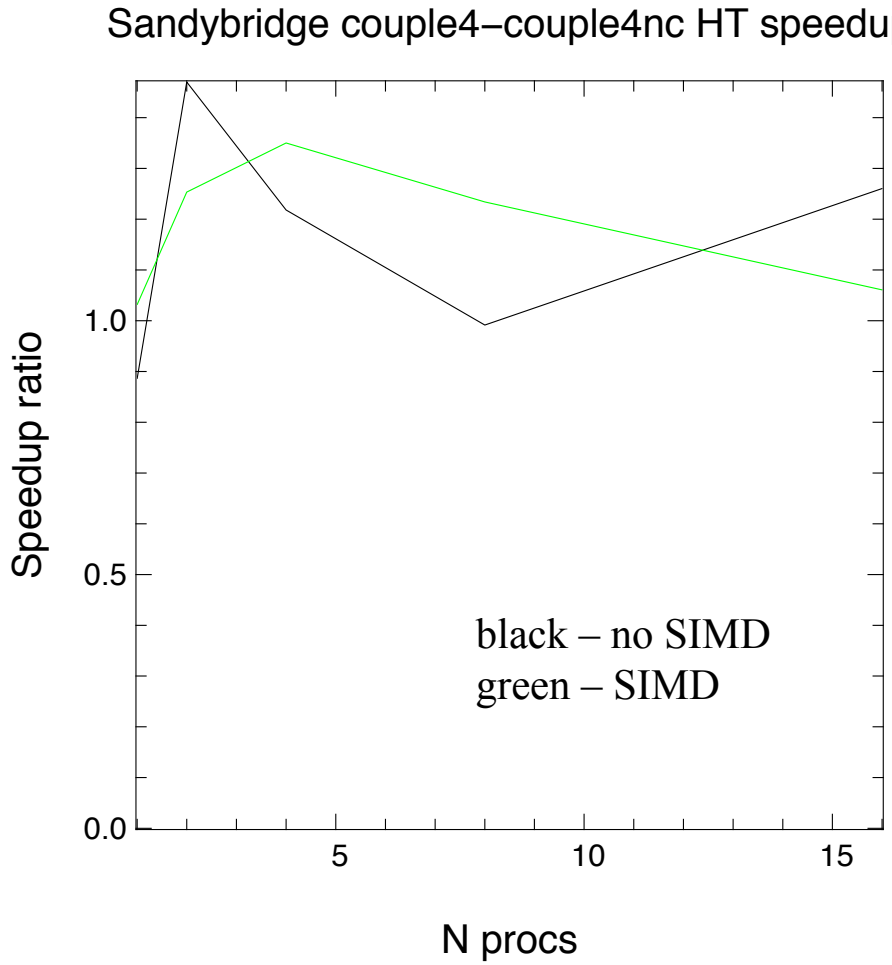



The Intel compiler (icc) cannot (so far) correctly distribute work across threads on the Magny-Cours.

gcc 4.6 can correctly distribute work, but is poor at SIMD vectorization.

The SIMD speedup for single threaded processes is over 2X. Similar speedups should occur with multiple threads if the problems with icc can be fixed.

# Hyperthreading increases the SIMD throughput of a Sandy Bridge node by 20-30%

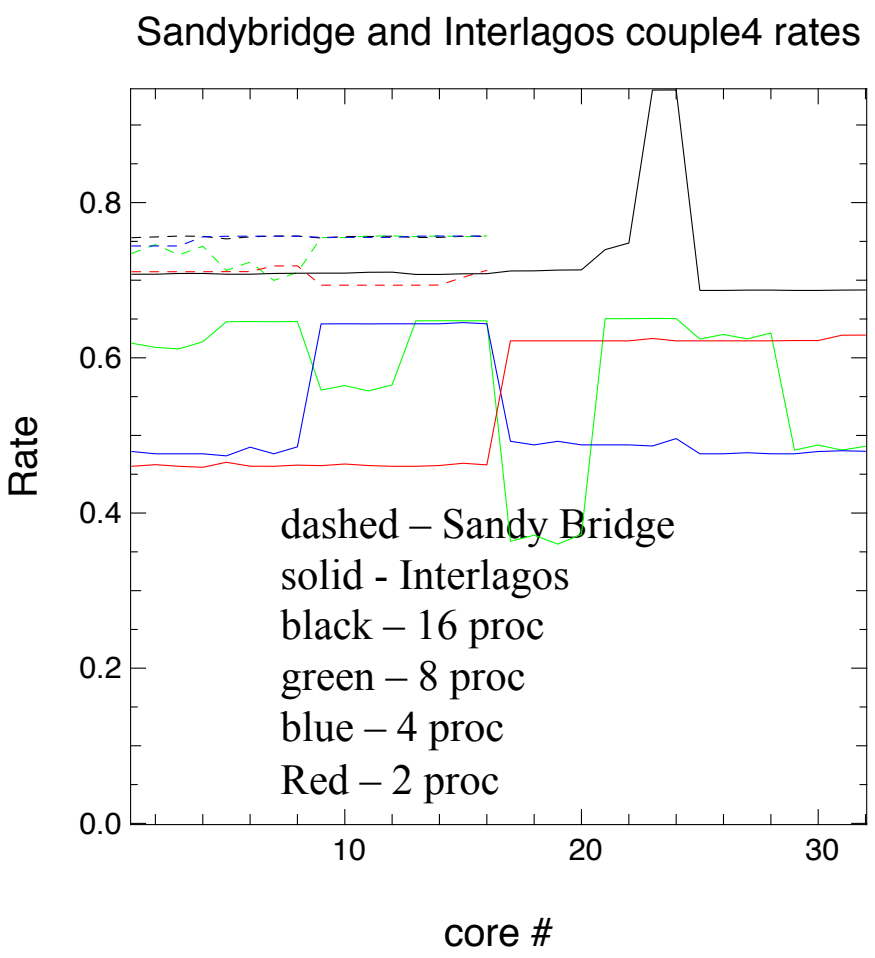



Instructions can be issued from both hardware threads without any context switching overhead.

Hyperthreading can improve throughput by hiding pipeline stalls and other types of latency.

Speedup occurs for both the SIMD and non-SIMD versions of couple4

# There is large thread-to-thread variability for couple4 on an AMD Interlagos node




The update rate is nearly the same for all threads on the Sandy Bridge system.

The rate varies by up to a factor of two from thread-to-thread on the Interlagos node.

Threads in the same process have the same performance.

The work per core on the Sandy Bridge and the work per floating point unit on the Interlagos was the same in all cases.

Thread binding to cores may not be working well on the Interlagos node.

## A resiliency strategy is important when running on large systems

- Large computer systems use reliable components, but they have a very large number of them.
- System hardware and software problems inevitably interrupt codes, like pF3D , that must run for many days on 32k or more Cielo cores.
- pF3D relies on periodically writing the state of the simulation to disk to permit recovery from system errors and to allow a run to resume in the next batch time slot.
- The 32k core simulation we recently completed on Cielo used 93 days to make 51 days worth of progress. This means that roughly 60% of the time spent on the simulation made progress.

## Checkpoints can consume a significant amount of time

- The 32k simulation could read or write a checkpoint in roughly 30 minutes.
- A 24 hour batch time slot performed one restart and wrote 3 dumps if the system performed well. Checkpoint I/O consumed roughly 10% (2 hours out of 24) of the wall clock time of our run in the absence of any errors.
- When a simulation crashes, it must "rewind" to the most recent checkpoint. During our runs, a checkpoint might be up to 7 hours old.
- The parallel file system was responsible for over half the interruptions of our simulation. Node and interconnect failures were responsible for most of the other interruptions.

## Checkpoints written to disk might be problematic for a run twice the size of the current simulation

- Scheduled I/O would take 4 out of 24 hours.
- Running on 96-128k cores will increase the frequency of node and interconnect failures
- The switch to Lustre may greatly decreases the frequency of I/O errors based on experience with other systems.
- If the MTBI winds up being the same, we could complete the simulation with reasonable efficiency.
- We are investigating a way of reducing the time spent on checkpoint I/O.

## SCR writes most checkpoints to RAM disk and reduces the time spent on checkpoint I/O

- SCR can write a checkpoint to RAM disk in a few seconds for our proposed large simulation.
- SCR writes a checkpoint to the file system once per batch slot.
- The checkpoint I/O time should drop from 4 hours to 2 hours when using SCR.
- SCR checkpoints are so fast that the can be made every time step. When a jobs crashes, the last checkpoint will be no more than 30 minutes old rather than up to 7 hours.
- SCR can restart a job from RAM disk within the same moab script if a few extra nodes are allocated, avoiding the need to flush a checkpoint to disk before the end of the 24 hour batch slot.

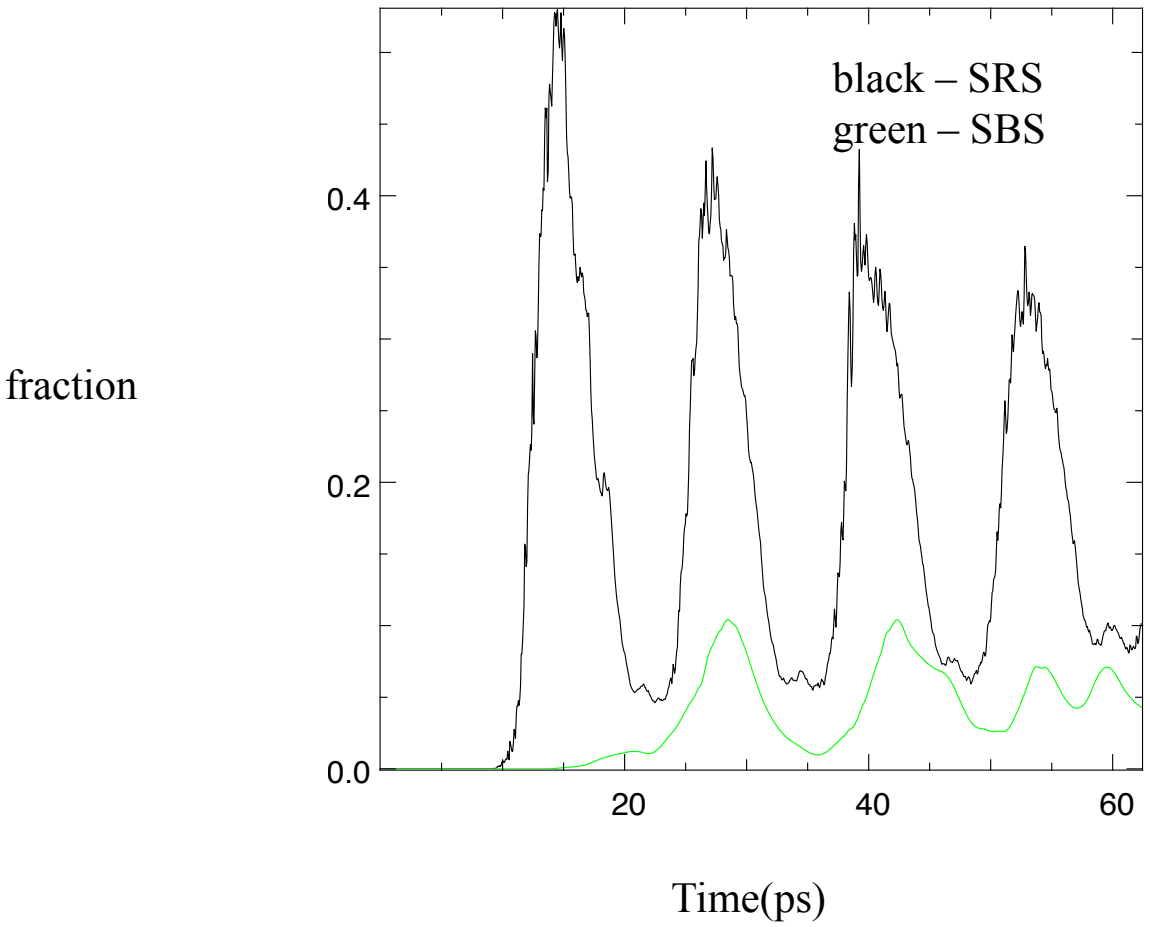
## SCR testing on Cielo is beginning

- SCR works best if there is a high probability of successfully writing the checkpoint to the file system at the end of a batch run.
- The frequent file system failures on Cielo would negate much of the benefit from SCR.
- We are waiting for the Lustre file system to be available before attempting to evaluate the benefit of SCR on Cielo.
- In earlier runs on an Opteron Infiniband cluster, SCR improved the throughput of pF3D by 50%.

## pF3D simulations on Cielo are helping us understand NIF experiments

- The simulations have bursts of backscattered light with periods of about 10 ps. Detectors cannot resolve this time scale, so time averaged backscatter levels are used for comparison.
- The simulations have SRS that is several times stronger than SBS. The simulation was run at a time when SRS was rapidly increasing and SBS was rapidly decreasing in the experiment.
- The backscatter fractions are somewhat less than the observed values. Future simulations may increase the interaction volume which may increase the SBS levels and perhaps the SRS levels.
- There is interesting spatial structure in the simulations that may help us understand the competition between SRS and SBS.

# The backscattered light comes in bursts



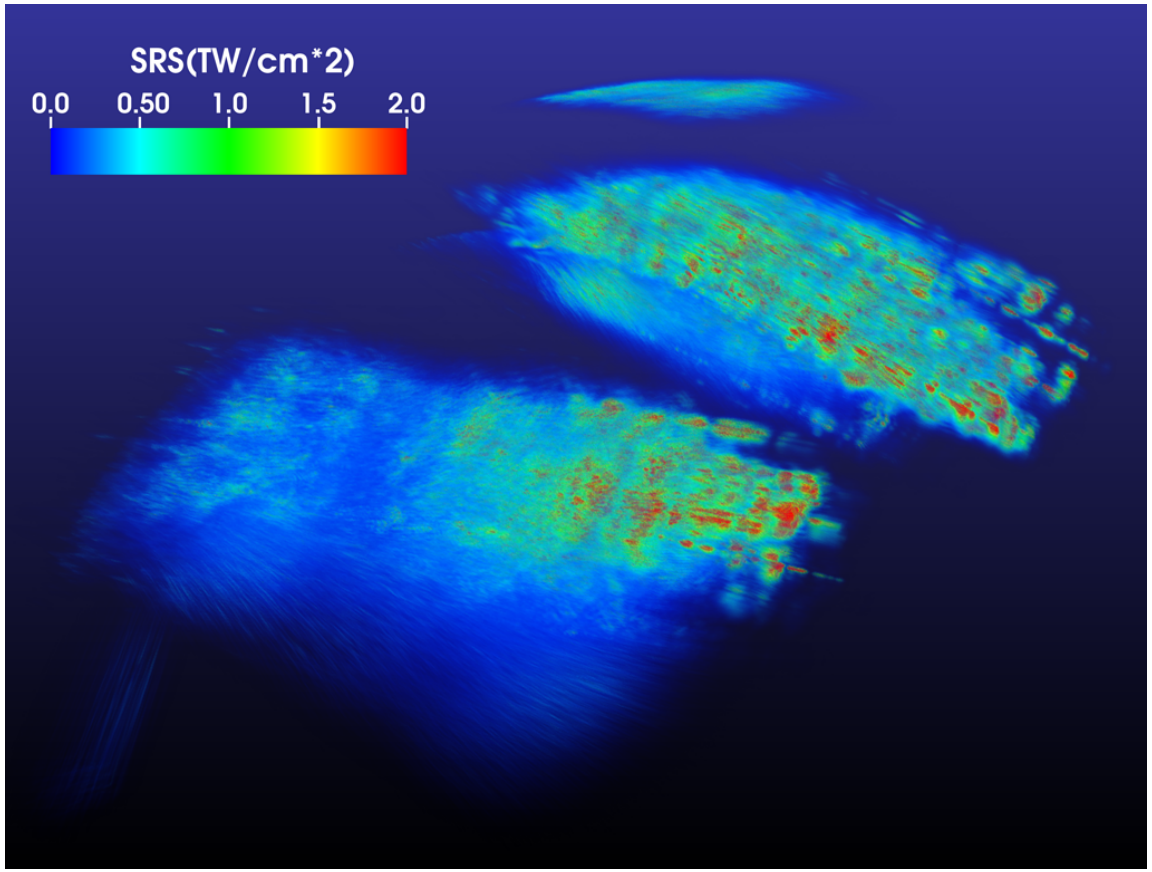

SRS and SBS both come in bursts.

SRS is stronger than SBS.

The SBS appears to be settling into a low variability state at the end of the run.

VOLVIZ at last SRS peak and at final SBS peak??

# The SRS light has a complex spatial structure




The SRS backscattered light is shown using a volume visualization.

The SRS gains strength as it moves from left to right.

There is a lot of small spatial scale structure in the SRS.

# pF3D has run several large simulations on Cielo and is undergoing preparation for many-core machines

- pF3D simulates the interaction between an intense laser beam and the ionized gas through which it travels.
- A recent simulation on Cielo ran for several months (7 CPU millenia) on 32,768 cores and had over 220 billion zones.
- pF3D performs well on Cielo.
- Custom domain to interconnect location software is being developed to better exploit toroidal interconnects.
- OpenMP parallelism is being added to pF3D to better exploit many core chips.
- More efficient checkpointing schemes are being added to pF3D to improve our resiliency.