

Large-Scale Simulations with Miranda on BlueGene/L

**William Cabot, Andrew Cook
& Charles Crabb**

Visualization

**Dan Laney, Mark Miller, Hank Childs,
Randy Frank, Liam Krauss**



**Lawrence Livermore
National Laboratory**

**BlueGene/L Workshop, Reno
14–15 October 2003**

UCRL-PRES-200327

This work was performed under the auspices of the U.S. Department of Energy by the University of California, Lawrence Livermore National Laboratory under Contract No. W-7405-Eng-48

Overview



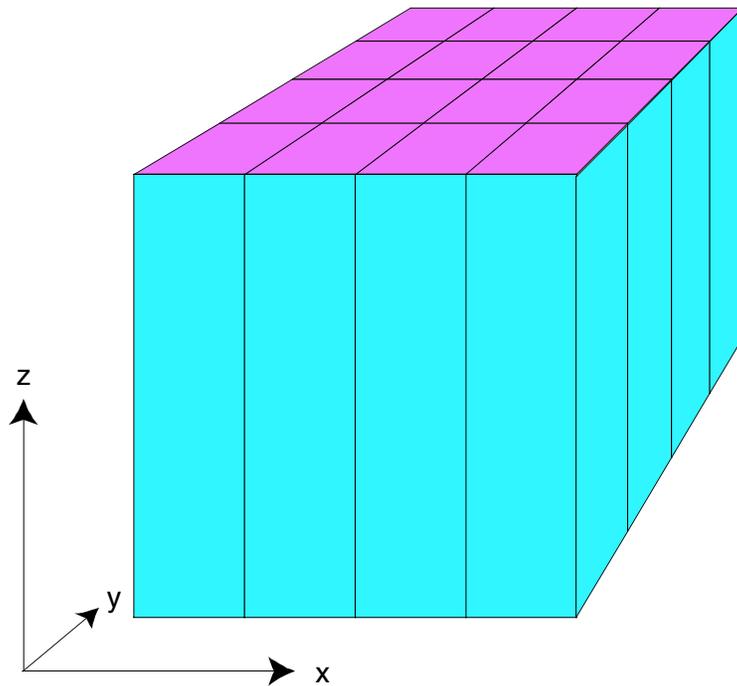
- **Introduction to the Miranda code**
- **Prior simulations on ALC & MCR (~2K CPUs)**
- **Projected large runs on BG/L (~100K CPUs)**

The Miranda Code



- **Miranda is a research hydrodynamics code ideal for simulating **Rayleigh-Taylor** and **Richtmyer-Meshkov** instability growth**
 - **Fortran 95, MPI**
 - **incompressible and compressible**
 - **explicit time solution; Poisson solve for incompressible**
 - **Eulerian (fixed), Cartesian mesh**
 - **high-order accurate derivatives**
- **The code uses 10th-order compact (spectral-like) or spectral schemes in all directions to compute global derivative and filtering operations**
- **Data is transposed to perform sparse linear (pentadiagonal) solves and FFTs, which requires lots of MPI communication (MPI_ALLTOALL)**

Miranda data layout

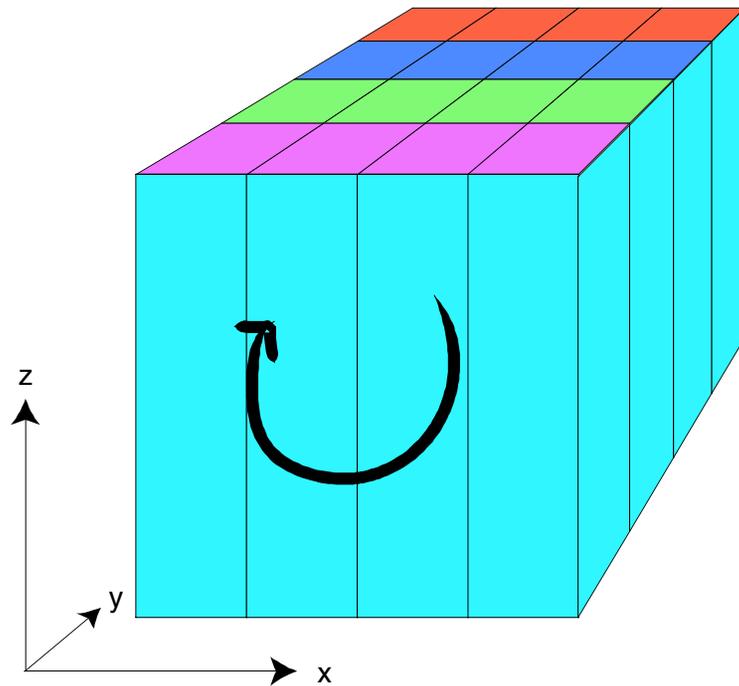


- 2D data layout on processors
- Derivatives and filters are *global operations* that are performed in plenum directions



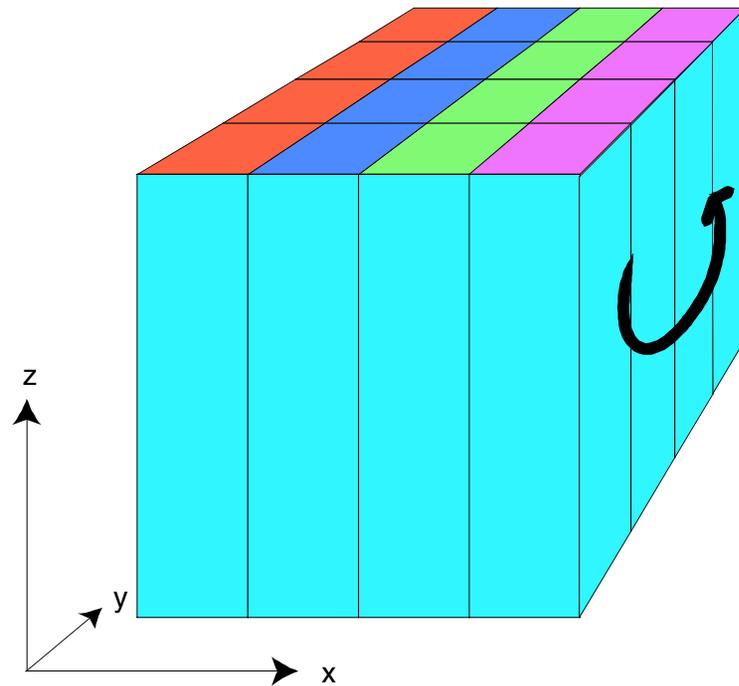
“Parallel” data transposition in Miranda

- Transpose data in x



- MPI_ALLTOALL on all x communicators

- Transpose data in y

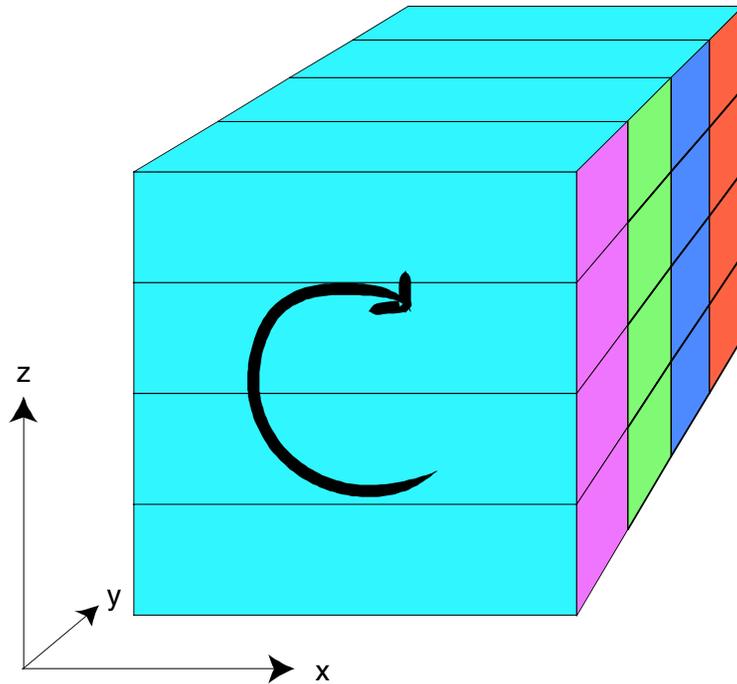


- MPI_ALLTOALL on all y communicators



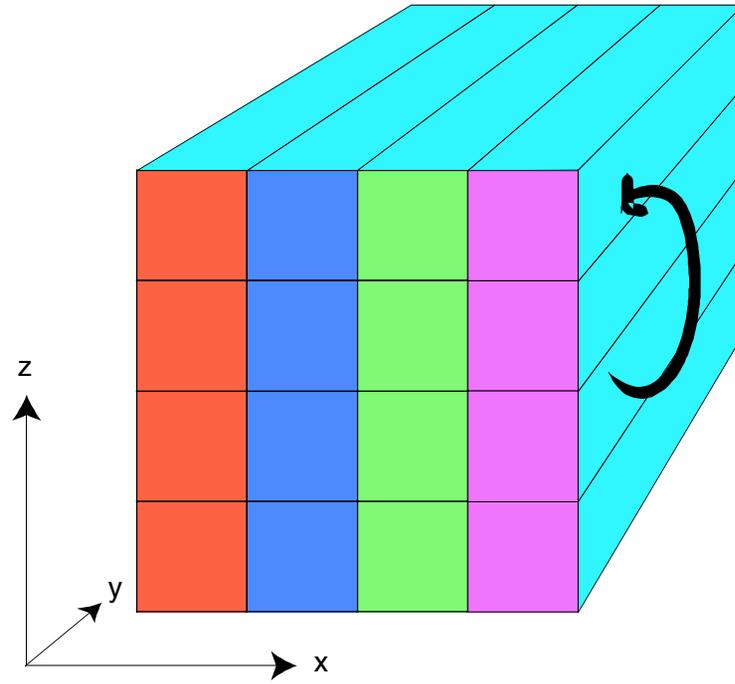
“Parallel” data transposition in Miranda

- Transpose data in x



- MPI_ALLTOALL on all x communicators

- Transpose data in y

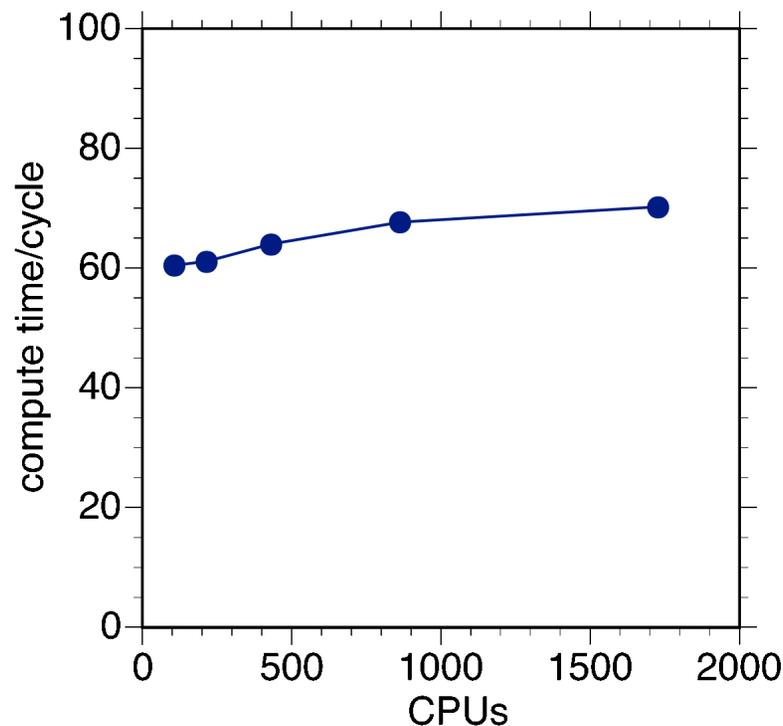


- MPI_ALLTOALL on all y communicators

The Miranda Code, continued



- Communication is mostly **synchronous**
 - MPI_ALLTOALL on $\sqrt{N_p}$ x,y comms for global operations
 - some MPI reductions and broadcasts for statistics
- Current I/O model is $n \rightarrow n$: each CPU writes/reads its own restart and graphics data files (lots of files!)



**Good scalability was found on
ALC / MCR up to 1728 CPUs**

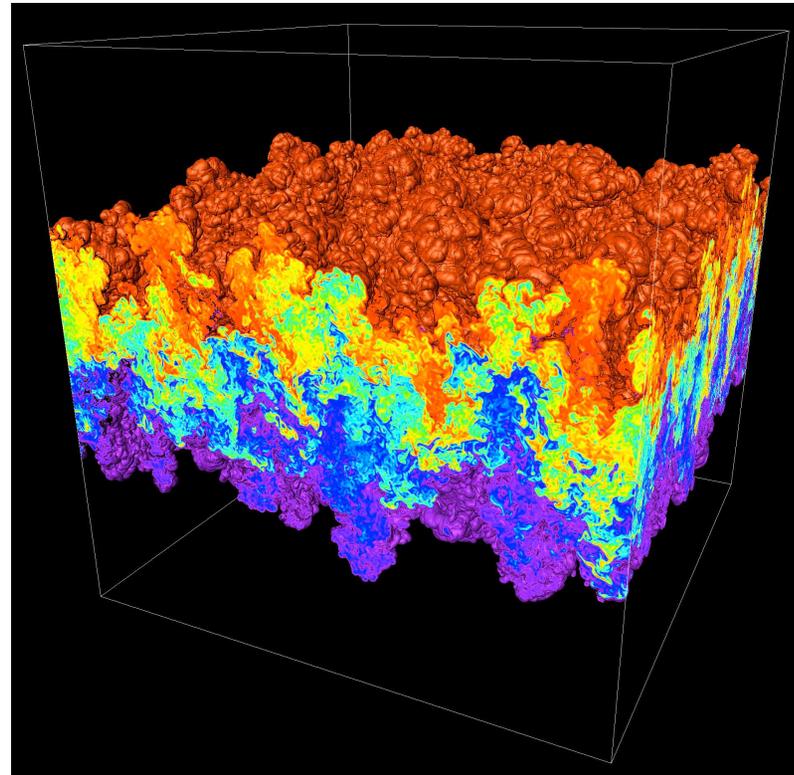
Communication scaling on MCR
for fixed workload per CPU

Large-scale simulations on ALC and MCR



- We have used nearly whole-machine runs on **ALC & MCR** to simulate fundamental turbulence problems using the **Miranda** code

Rayleigh-Taylor
instability,
density



heavy
mixed
light

- **ALC and MCR are sibling Linux clusters with ~1000 Intel P4 nodes with dual 2.4 GHz processors**
- **Lustre file system, DDN servers, 24/84 TB available storage**

MCR/ALC Run Stats



- **Rayleigh-Taylor Instability** simulation cases used
 - ◆ 720 x 720 x 1620 grid points on 810 nodes (1620 CPUs)
 - ◆ 1152 x 1152 x 1152 grid points on 864 nodes (1728 CPUs)
- The simulations each required
 - ◆ 30k – 40k cycles
 - ◆ 20 – 30 CPU days of “quality run time”
- Data generated (written to disk) for each case:
 - ◆ Graphics: 1.3 million files, 20 – 30 TB (100% archived)
 - ◆ Restarts: 0.3 million files, 20 – 30 TB (25% archived)
- Dealing with the large amount of data storage needed to make movies and for future data mining has presented a challenge



Rayleigh-Taylor Instability FAQ

What is it?

- **Rayleigh-Taylor (RT) instability** occurs when a perturbed heavy-light fluid interface experiences a *steady* acceleration toward the light fluid
- **Richtmyer-Meshkov (RM) instability** occurs for an *impulsive* acceleration

Why do it?

- RT and RM instabilities affect the performance of ICF capsules
- Numerical simulations complement experiments
- Very large simulations are needed to achieve a sufficiently wide range of scales in space and time to represent accurately the **turbulent** flows of interest

What's the cost?

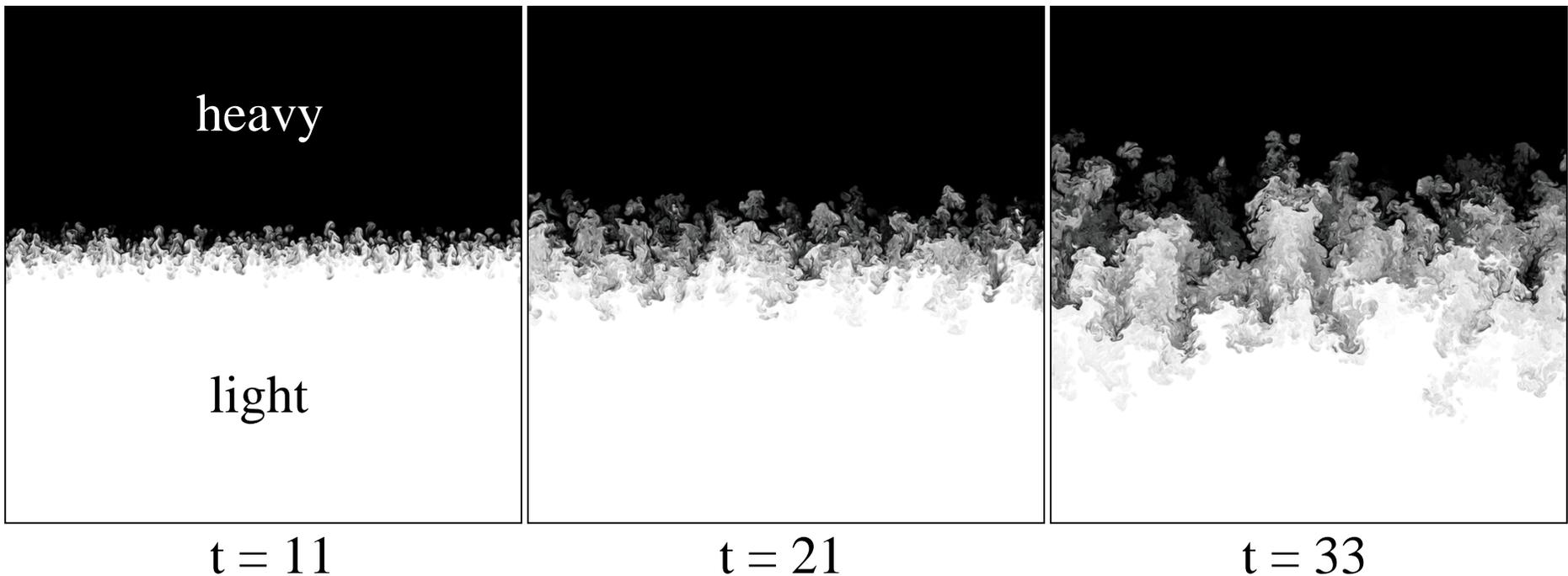
- **A lot! For 3D hydro simulations:**

memory \sim resolution³; run time \sim resolution⁴



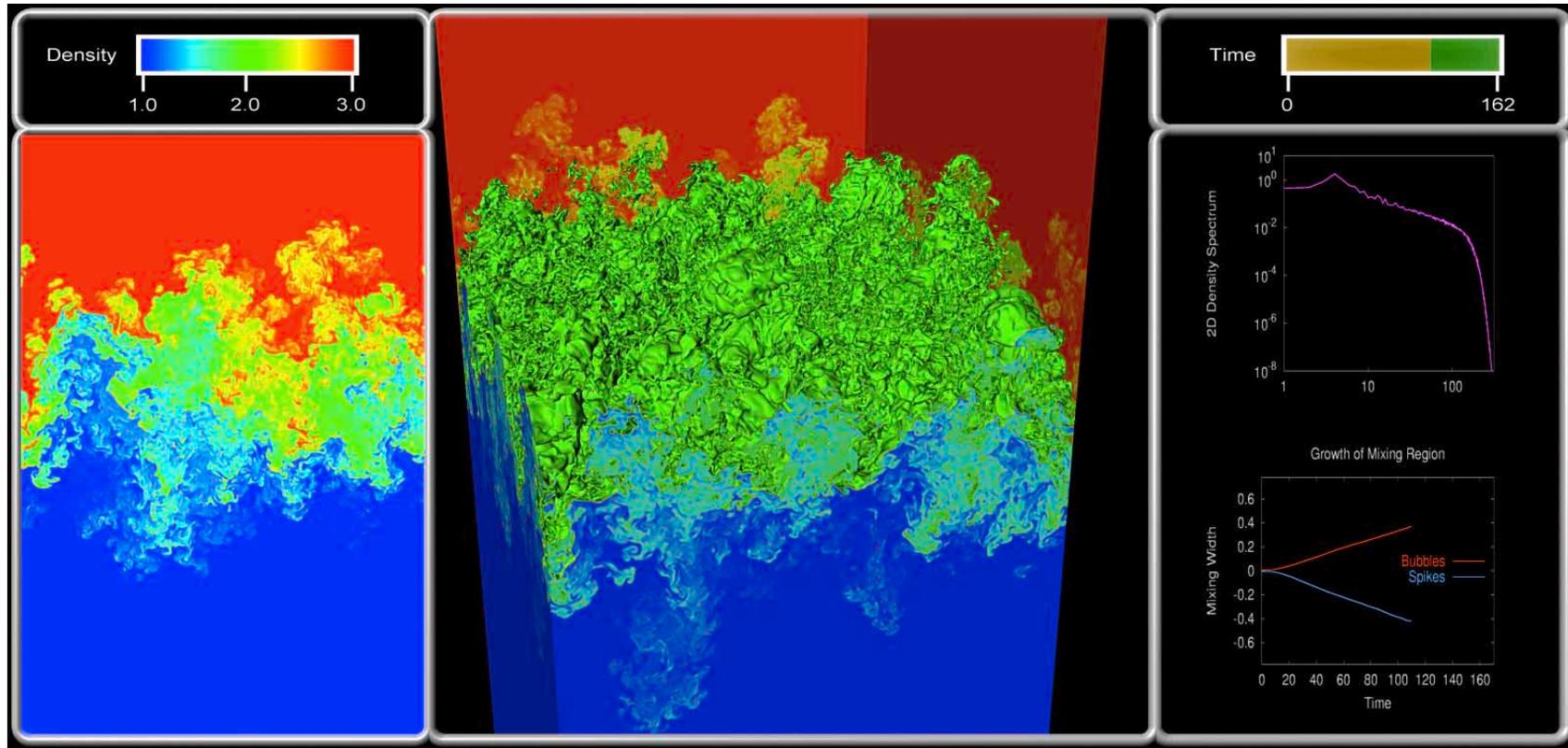
Rayleigh-Taylor Instability Simulation Setup

- Planar, miscible (with Fickian diffusion), incompressible
- 3:1 density ratio in uniform gravity with a slightly diffuse interface initially perturbed with small-scale fluctuations
- Periodic horizontal and impermeable vertical BCs
- Require both horizontal and vertical flow scales \ll domain size



1152^3 run: Density (side-view slices)

RT – The Movie



Projected Large Runs on BG/L



- Our objective on BG/L:
 - run the *largest* 3D RT / RM turbulence simulations
 - establish late-time asymptotic growth *in the fully turbulent regime*, not yet accomplished with $(1K)^3$ runs
- Memory on BG/L constrains the size of the largest runs
 - Miranda uses about 1 TB of memory per $(1K)^3$ mesh
 - BG/L is currently projected to have 32 TB memory available, which limits us to a $(3K)^3$ mesh problem (unless Miranda is put on a serious diet...)
 - This is $(3072/1152)^4 = 50x$ previous run, $\sim 100k$ cycles
- Disk space: BG/L will have 400TB (5x MCR, 20x ALC)
 - potentially $(3072/1152)^{3-4} = 20-50x$ previous run \rightarrow 1PB
 - data archiving and in situ processing required

Projected Large Runs on BG/L: Scenarios



<u>grid</u>	<u>memory</u>	<u>nodes</u>	<u>CPU/node</u>	<u>CPUs</u>	<u>grid/CPU</u>	<u>run time*</u>
3072 ³	27 TB	64K	1	256 x 256	12 x 12 x 3072	40 CPU days
3072 ³	27 TB	64K	2	512 x 256	6 x 12 x 3072	20 CPU days
2880 ³	22 TB	64800	2	360 x 360	8 x 8 x 2880	15 CPU days

2560 ³	16 TB	64K/32K	1/2	256 x 256	10 x 10 x 2560	20 CPU days
2048 ³	8 TB	64K/32K	1/2	256 x 256	8 x 8 x 2048	8 CPU days

* lower limit, optimistically estimated using perfect scaling from MCR

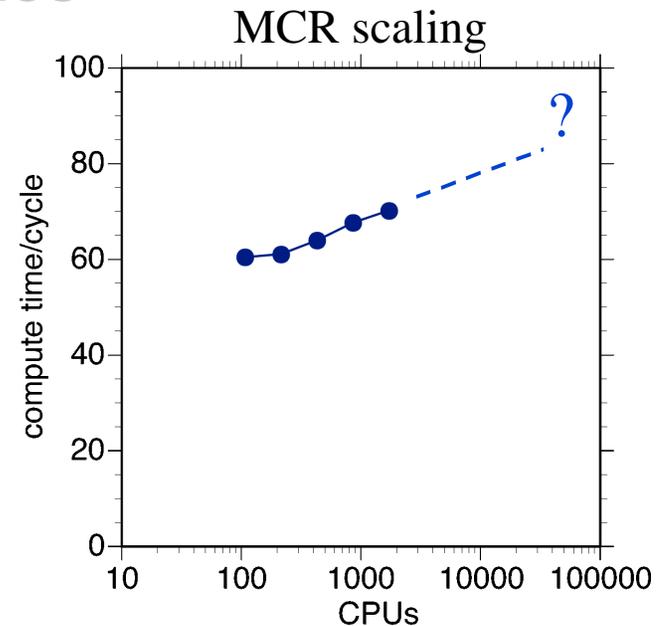
- ◆ Competition: 4K x 2K x 2K RT simulation performed on the Earth Simulator (Sakagami et al. 2002)

Porting and Performance Issues



Performance

- **Communication scaling for ~100K processors?**
 - test runs on > 10K CPUs needed to predict scaling!
 - early runs on 1/4 or 1/2 machine are required
- **Processor speed on BG/L is comparable (?) to MCR**
 - 2.8 (PPC440) v's 2.4 (Pentium 4) GF/s/CPU *for floats*; but for doubles and complex doubles, etc.?
 - use one or two CPUs per node?
- **Optimized linear solvers and FFTs, optimized data structure**
 - sacrifice portability for improved performance?
- **Use simulator / prototype for compiler issues and tuning**





Porting and Performance Issues , continued

Memory

- The relatively small amount of memory limits the size of the problems we can perform; almost all nodes are required for large (record-breaking) turbulence simulations
- We can slim down the Miranda code to use less memory, but only so much...

Data management

- Good data storage infrastructure; smarter data management
 - vast, fast disks & storage
 - compressed / reduced data archiving; in situ / on-the-fly statistics and graphics processing
 - Will $n \rightarrow n$ I/O model work?

Run-time stability

- Reliability and fault tolerance of system
- Effective checkpointing; redundancy?