PECOS

Predictive Engineering and Computational Sciences

Petascale I/O using HDF-5

Benchmarks and design

Nicholas Malaya, Karl Schultz, Robert Moser

Institute for Computational Engineering and Sciences
The University of Texas at Austin

August 4th, 2010

Outline

Simulating Turbulence

Turbulence

- Ubiquitous phenomena
  - plane wing, pipe flow, supernova, coffee and cream
  - 20% of annual USA energy consumption is used for transportation

Direct Numerical Simulations of Turbulent Flow

- Resolves all scales – no modeling
- Higher reynolds number requires greater number of points
  - Cost increases as Re^3 – need big machines!
- Diverse set of problems being simulated
  - channel flow, stratification, homogeneous turbulence ...
  - University of Texas (R. D. Moser), Georgia Tech (P.K. Yeung), UWash (Riley), SDSC (Majumdar, Pekurovsky)
- Objective: release portable, open source library for canonical flows

... in addition to doing science

DNS is a postprocessing problem

Postprocessing Requirements

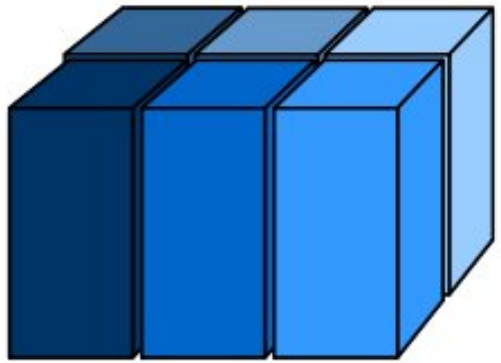
- I/O is not just for restarting simulations.
- Primary task of DNS is to compute statistical diagnostics
  - Fields are used like an experimental facility to “measure” any quantity
    - “Truth Data”
  - As a result, fields are archived and accessed over decades, by various research groups
    - Portability Issues
  - Significant fraction of simulation is used on postprocessing
- Increasingly a requirement of open scientific research
  - We can save everything

This is a substantial metadata requirement.

Simulation Requirements

DNS Problem size

- Structured grid
- Mesh: 8192^3 (or equivalent)
- "Pencil" decomposition
  - "Long" in y
  - Segmented in x & z
- Target: 100k - 250k+ cores on Blue Waters



I/O Requirements

- $8 \times 3 \times N_y N_x N_z \approx 10$ TB per file
  - Can grow as large grow as 50 TB
- Write frequency approximately one per wall clock hour
  - Writing ≈ 500
  - Archiving ≈ 100
- Minimize cost towards simulation runtime ($\approx 10\%$)

Old Method

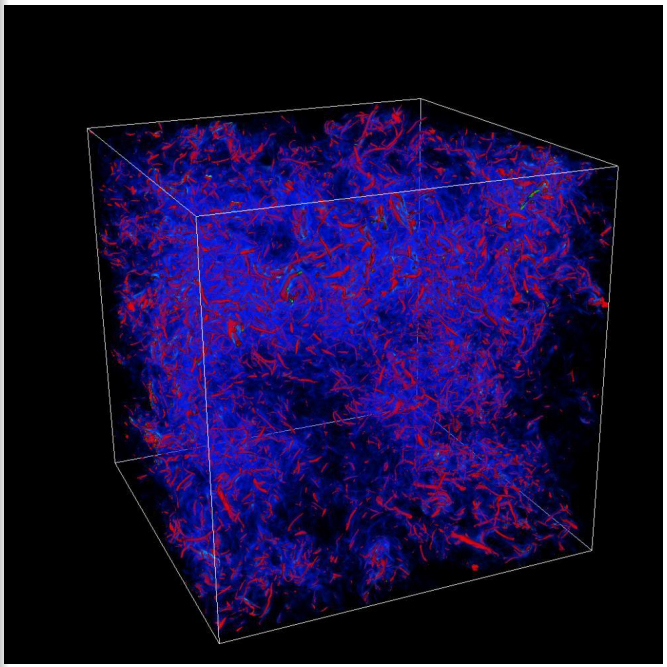
Many File I/O

- Every MPI task writes a single file containing a portion of the decomposed solution (“Pencil”)
- Pros:
  - Parallelism
  - High performance
  - I/O code is simple
- Cons:
  - Huge number of files
  - No metadata
  - Difficult to read on different number of processors
  - `ls`,`rm`, archive, `bbcp`
  - Often requires additional post-processing

New Effort

Unified HDF I/O (ESIO)

- Single file using modern libraries for portability and metadata.
- Pros:
  - Single File Output
  - Decomposition Independent
  - Easily read by visualization software
  - Files are portable
- Cons:
  - Slower than many file I/O
  - Requires partitioning information (offset)
  - More tuning needed for performance
  - Massive file size
  - Requires HDF library support



Benchmarks

ESIO library

- ExaScale I/O
- Single file output, extensive metadata infrastructure
- C and Fortran bindings
- HDF “tuning” parameters
- Lustre “dials” (stripe count, stripe size)

POSIX

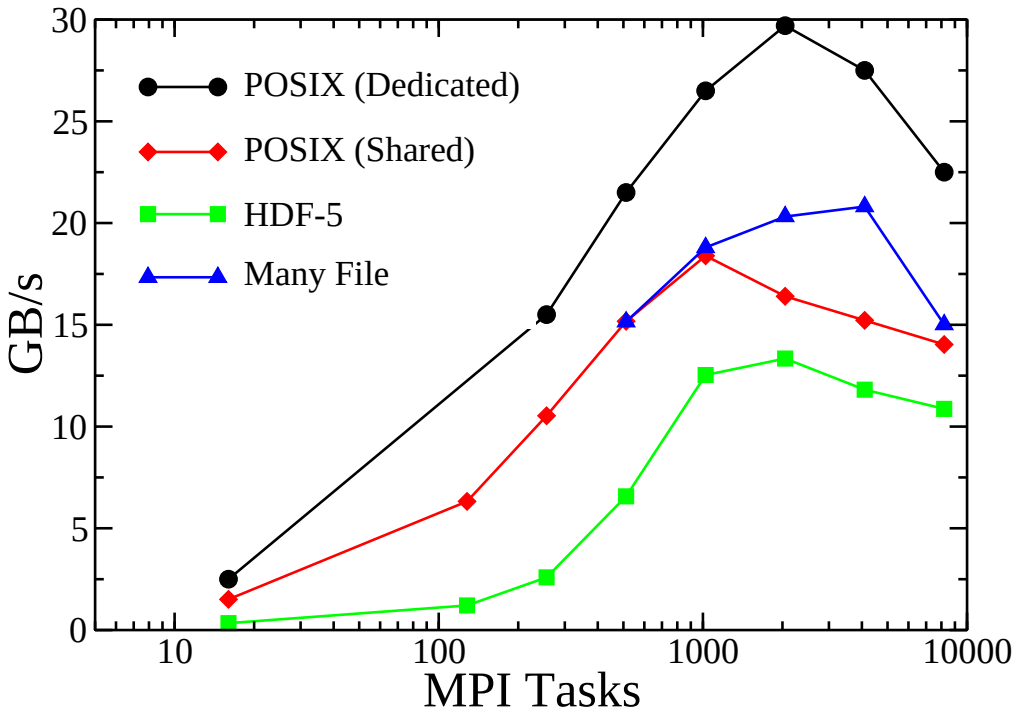
- “Quiet” system peak throughput is misleading
  - also harder to measure
- `fopen()`,`fseek()`,`fwrite()`
- Tuned for stripe count, stripe size, block size
- Single file, no decomposition (“cube”)

Benchmarking Method

- 20 MB write size per proc
  - “weak scaling” - on node performance and communication interconnects
  - largest output was 163 GB
- 10 samples per datapoint: Write speeds vary greatly
- Timing information from averaged `MPI_WTIME()`

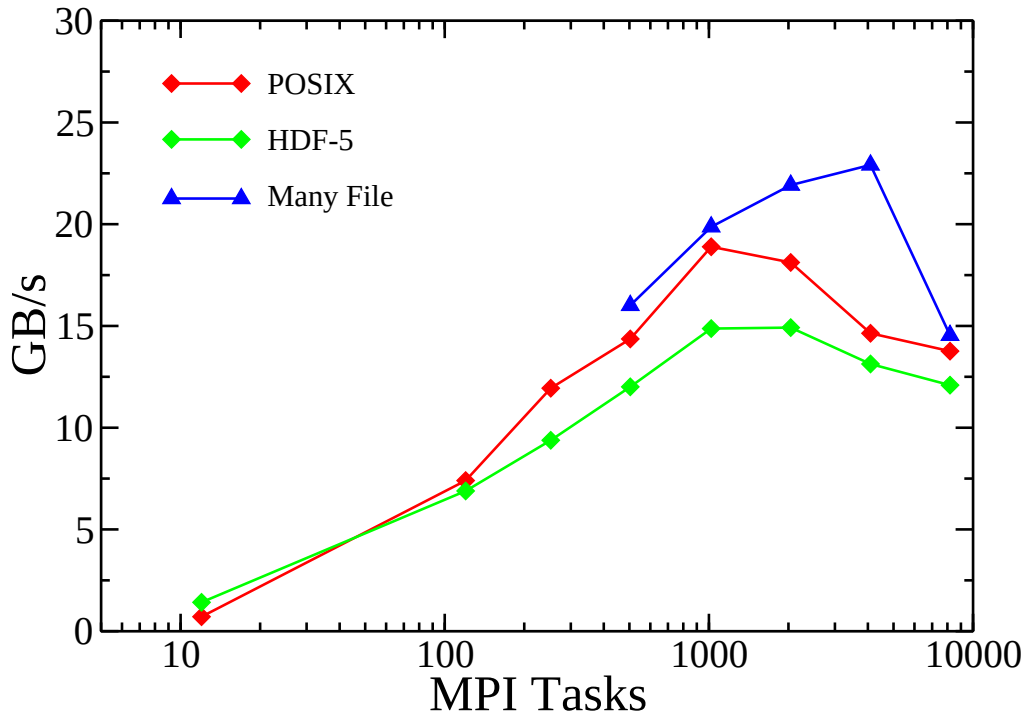
Write speeds on Ranger

- 30 GB/s peak throughput (dedicated)
- 18 GB/s production
- 13.34 GB/s HDF-5: 26.6% reduction



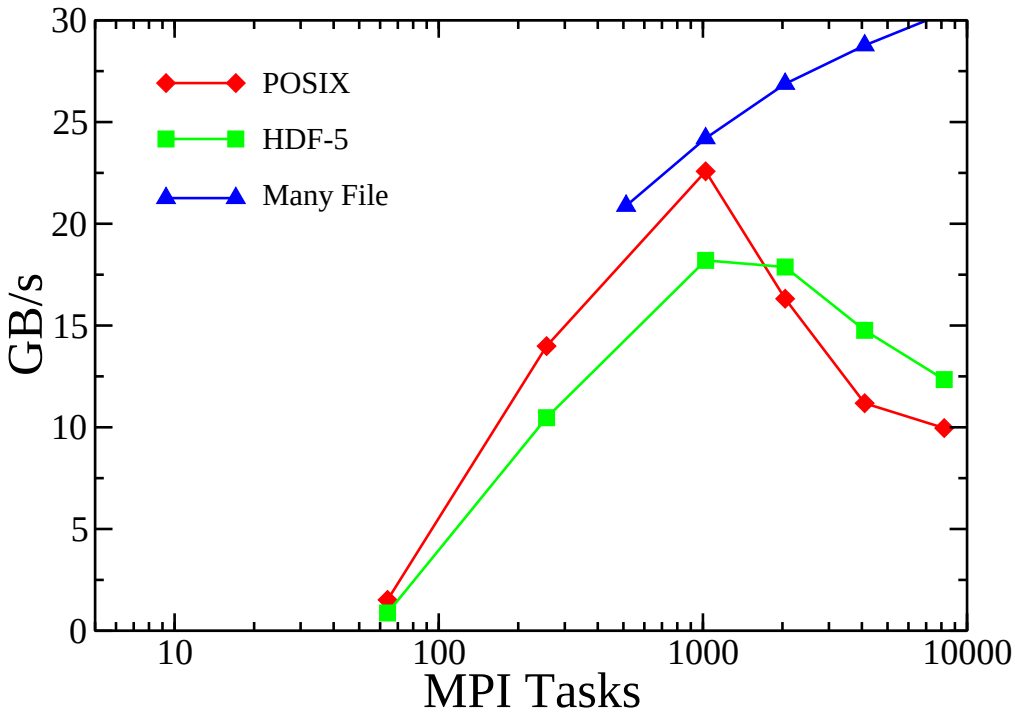
Write speeds on Kraken

- Did not measure dedicated write speeds (quoted 30 GB/s)
- “Relay” strategy: subsetting I/O



Write speeds on Intrepid (Blue-Gene/P Argonne)

- GPFS filesystem
- Not exposed for tuning
- Quoted peak speed of 65 GB/s



Tuning

- Varies by machine: Performing I/O operations at 4 MB on BG/p instead of 2 MB on Kraken. (`H5Pset_sieve_buf_size`)
  - consult your local supercomputing center
- `H5Pset_hyper_cache` – cache hyperslabs
- Look out for file-system contention. Consider relay strategy
- Set H5Pset_meta_block_size to I/O block size
- Performance still an issue: 20 GB/s speeds require $\approx$ 8 minutes for target file size
  - $\approx$ 10% per wall clock hour

Future Work

Conclusions

- Acceptable performance hit vs. barebones I/O schemes
- Optimizations can really improve performance
- Why 160 stripe count limit on Lustre?

Future Work / Interests

- Benchmarking at Extreme core counts – 100k+
- Open Release (LGPL)
- Further platform tuning (`FFTW_PLAN`)
- Asynchronous I/O (especially NCSA Blue Waters or ALCF Mira) RDMA
- Post-processing

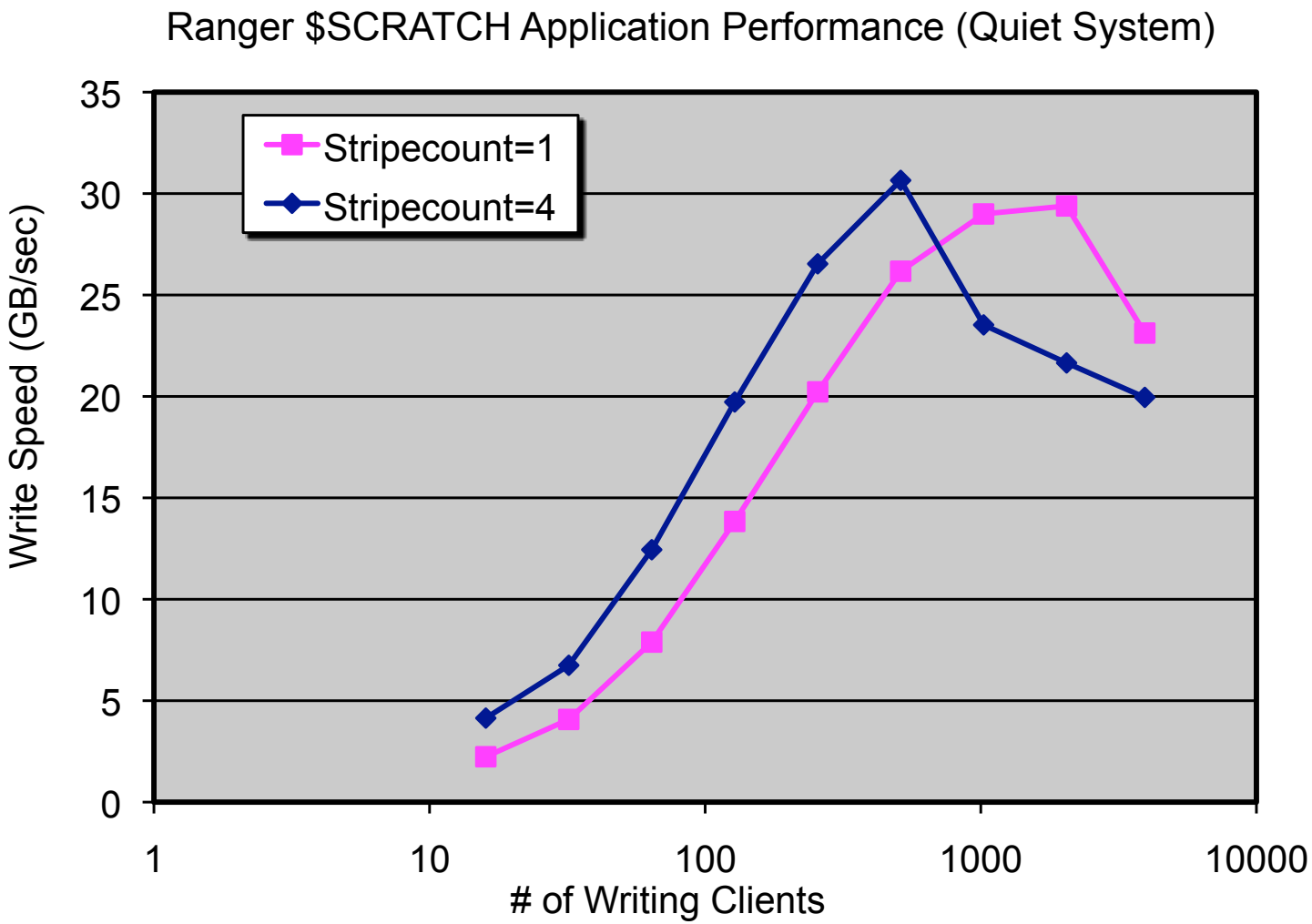
Thank you!

Questions?

This work supported by NSF PetaApps grant(s): OCI-0749223 and NSF PRAC Grant 0832634.

The author would also like to thank TACC, ALCF and NICS for tuning assistance.

Bonus Slide


Ranger $SCRATCH Application Performance (Quiet System)
Stripecount=1
Stripecount=4
Write Speed (GB/sec)
0
5
10
15
20
25
30
35
1
10
100
1000
10000
# of Writing Clients

Bonus Slide