# Lustre Lockahead: Early Experience and Performance using Optimized Locking

Michael Moore

CUG 2017 . CAFFEINATED COMPUTING
Redmond, Washington May 7-11, 2017

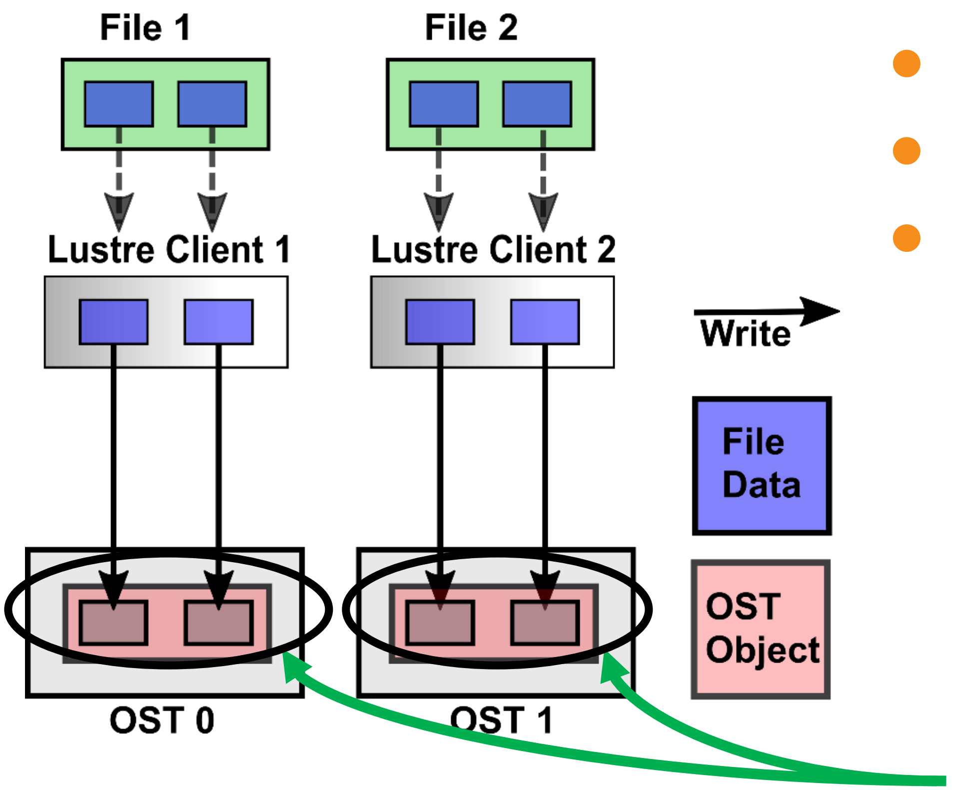
# Agenda

- **Purpose**
  - Investigate performance of a new Lustre and MPI-IO feature called Lustre Lockahead (LLA)
  - Discuss early experience and evaluate application for use with LLA
- **Topics**
  - Current Lustre locking and Lockahead details
  - LLA performance results
  - Application evaluation and tuning
- **Q&A**

# Acronyms

- **Lustre**
  - OSS (Object Storage Server)
  - OST (Object Storage Target)
- **I/O APIs**
  - POSIX (Portable Operating System Interface)
  - MPI-IO (Message Passing Interface I/O)
- **I/O Libraries**
  - HDF5 (Hierarchical Data Format)
  - pNetCDF (parallel Network Common Data Form)

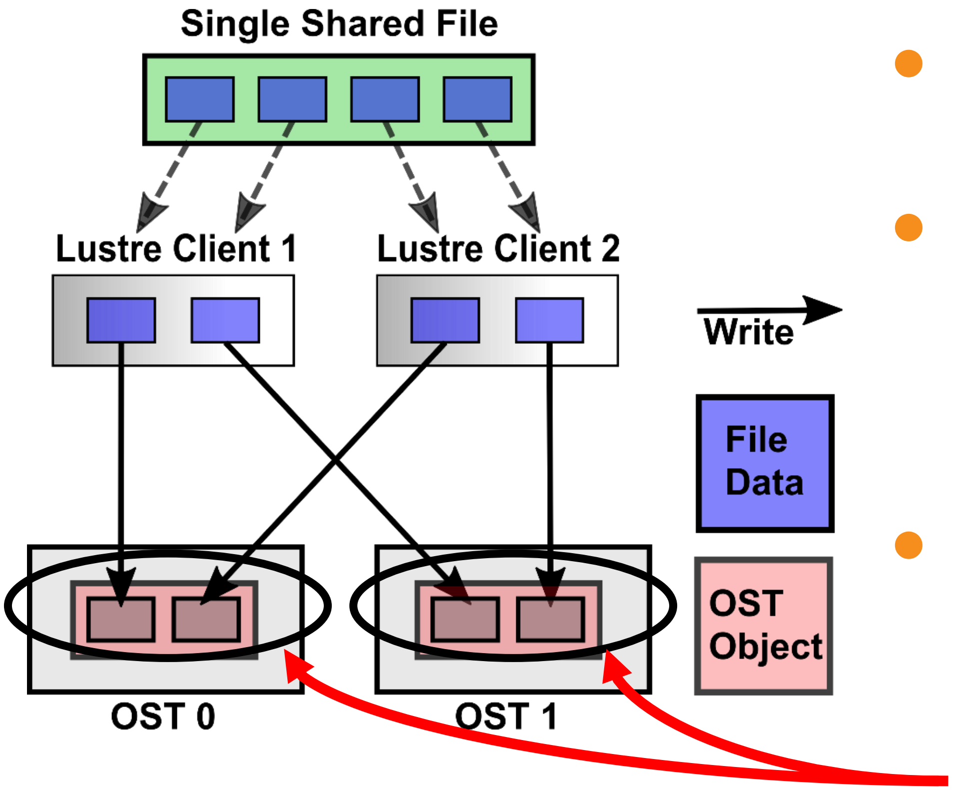
# File Per Process Access




- One file per MPI rank
- No lock contention
- Best performing I/O pattern for Lustre

Both clients acquire write locks with no contention

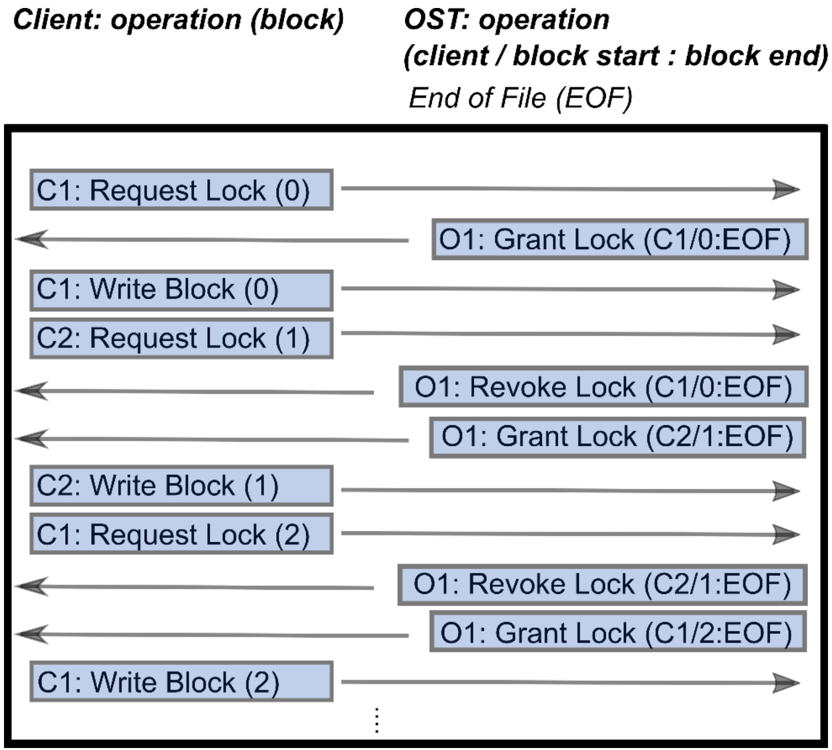
# Shared File Access




- One file accessed by all MPI ranks
- Multiple Lustre clients accessing a shared file requires locking between clients
- Lower performing compared to FPP

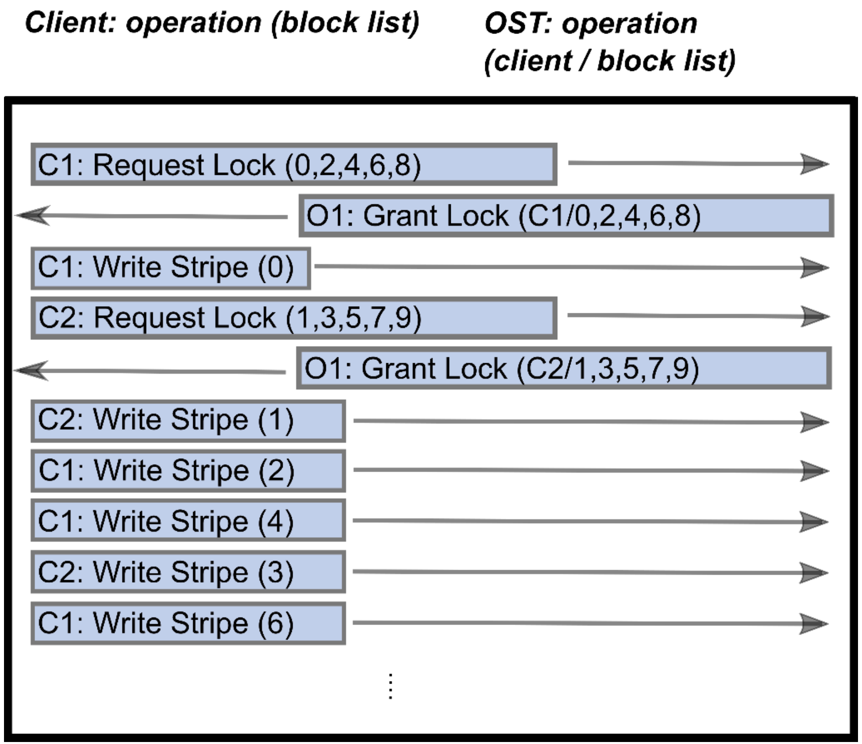
**Both clients accessing the same OST object cause lock contention**

# Lustre Locking Overview




- Default Lustre locking expands lock requests
- Lock expansion leads to lock contention due to false sharing
- Lock expansion does improve performance over no lock expansion

# LLA Overview

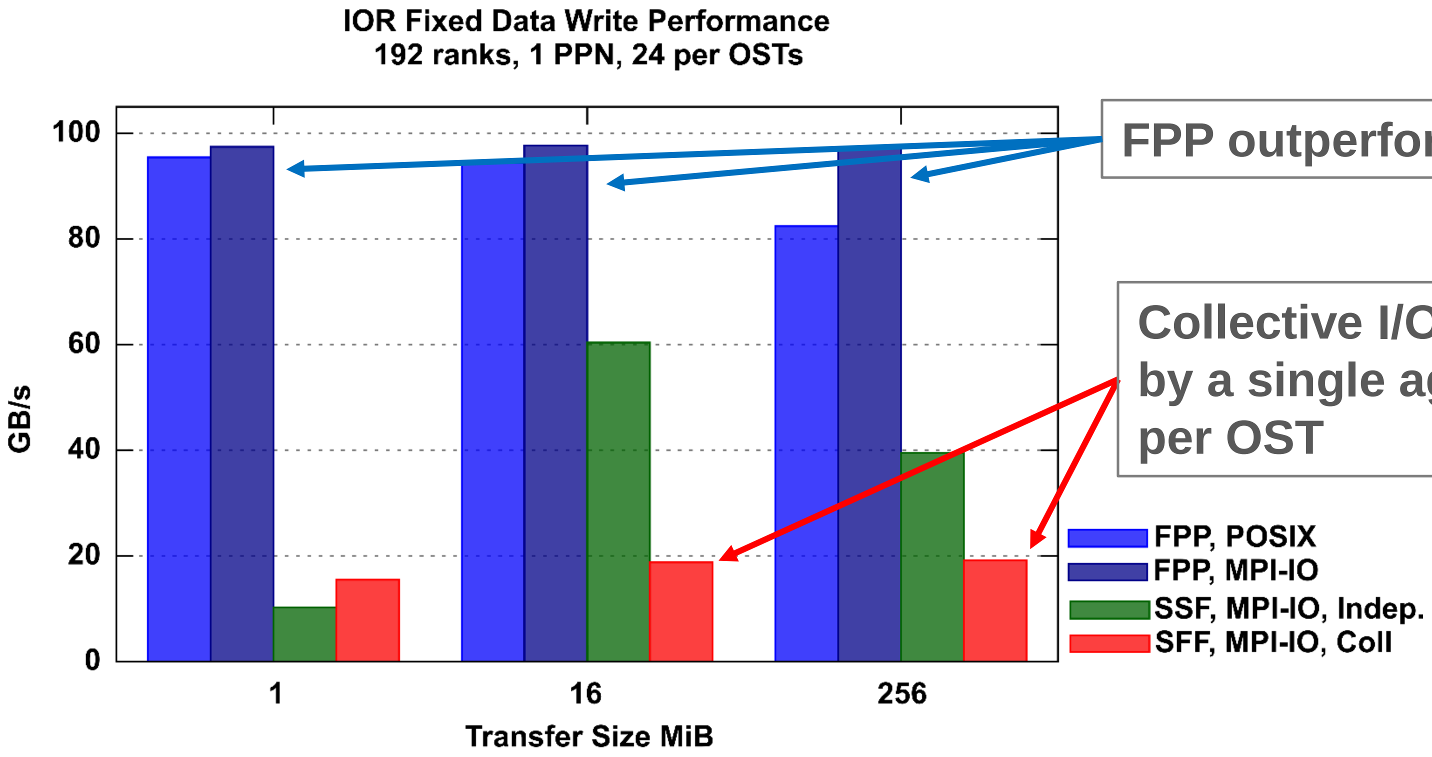



- No OSS lock expansion
- LLA requests multiple locks asynchronously to write requests
- Benefits
  - No false sharing
  - Lock acquisition is not part of write path

# LLA – Locking Ahead

- **Collective MPI-IO**
  - Writes are non-overlapping
  - Collective buffering allows MPI-IO aggregator ranks to know what file locations it will write
  - MPI-IO aggregator ranks can “lock ahead” – requesting locks ahead of where the MPI-IO aggregator rank is currently writing
  - MPI-IO aggregator ranks can request locks asynchronously to writes

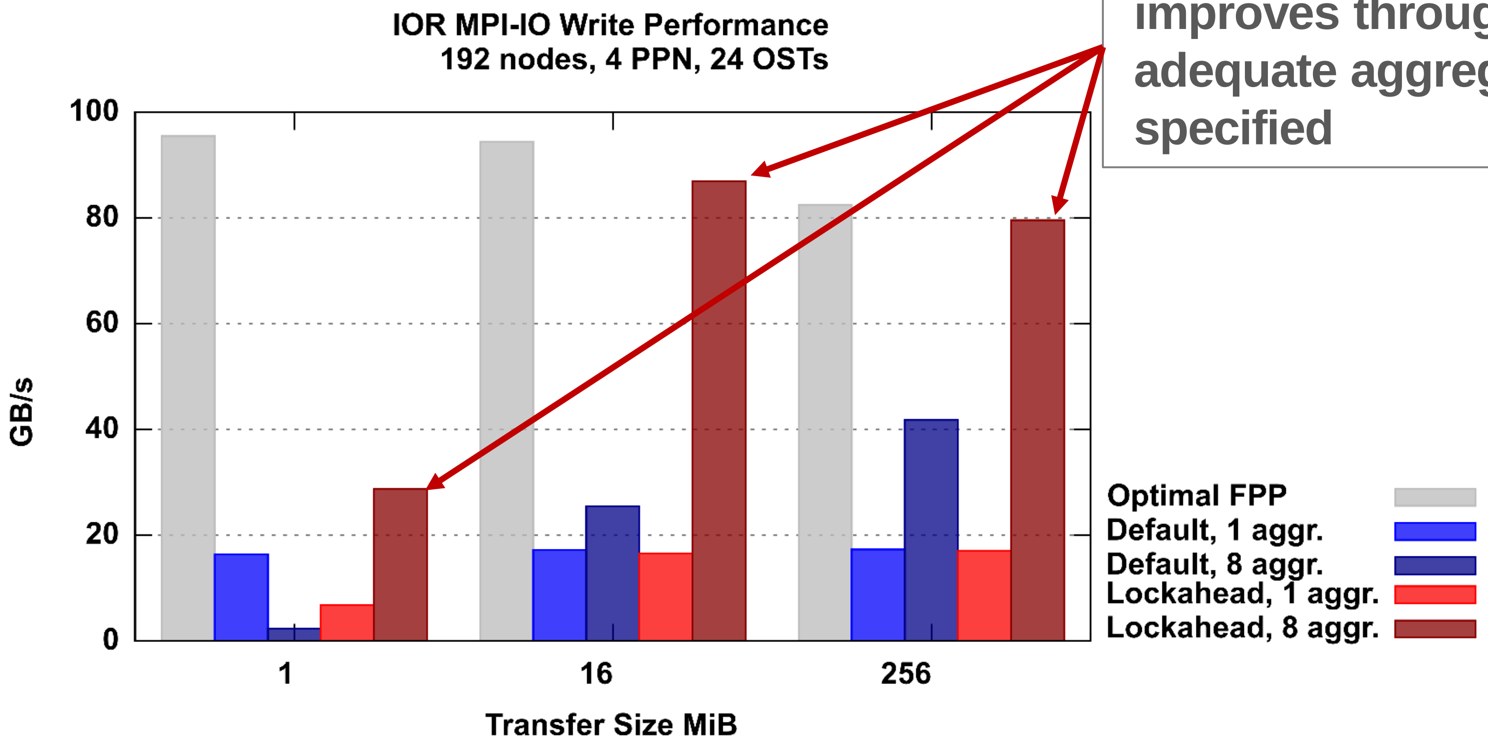
# IOR Performance, As Is




FPP outperforms all SSF

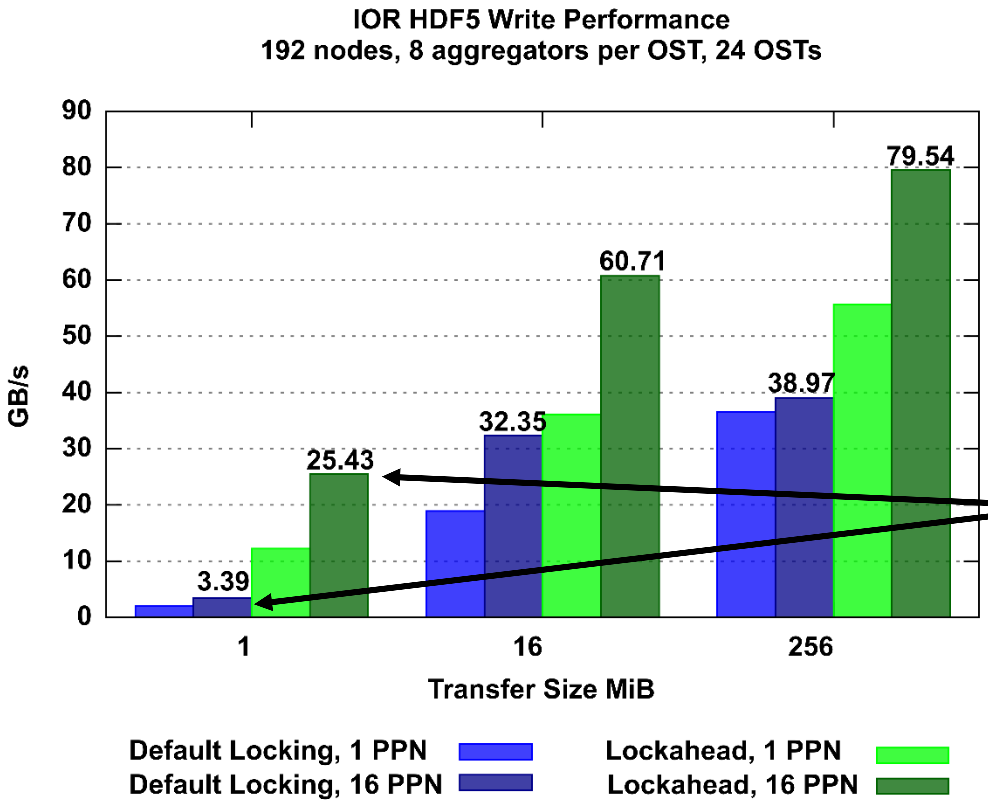
Collective I/O is bound by a single aggregator per OST

# IOR Performance, Collective MPI-IO




Lockahead dramatically improves throughput with adequate aggregators specified

# IOR Performance, HDF5

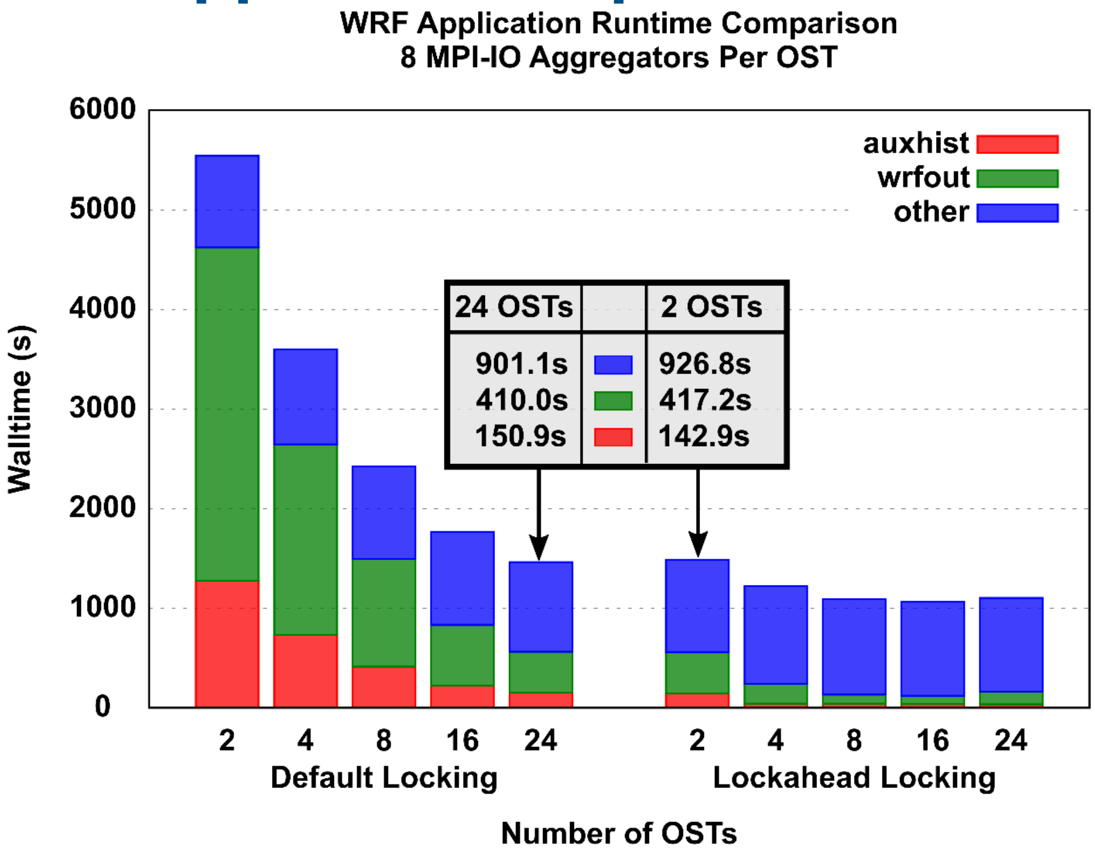



**1 MiB transfers show the overhead of Lustre lock contention; LLA provides a 7.5x improvement**

# Early Experience LLA Case Study

- **Weather Research & Forecasting Model (WRF)**
  - Study writes 1.1TB of data per job (restart and history files)
  - Job configured to use pNetCDF with collective MPI-IO
- **Enabling LLA for collective MPI-IO requires modifying the environment variable `MPICH_MPIIO_HINTS`**
  - Default Locking `"wrfout*:striping_factor=24"`
  - LLA `"wrfout*:cray_cb_write_lock_mode=2: \`
    `cray_cb_nodes_multiplier=8:striping_factor=24"`

# WRF Application Improvement




"other":
(*walltime – MPIIO write time*)

# Application Evaluation

- **Enable MPI-IO output for existing application**

```
MPICH_MPIIO_HINTS_DISPLAY=1
MPICH_MPIIO_AGGREGATOR_PLACEMENT_DISPLAY=1
MPICH_MPIIO_STATS=1
MPICH_MPIIO_TIMERS=1
```

- **Evaluate output**
  - File I/O details: Lustre striping, file size, MPI-IO call counts
  - Collective utilization and timing

# Application Evaluation, MPI-IO stats

- **WRF example of MPI-IO stats**

```
+--------------------------------------------------------+
| MPIIO write access patterns for wrfout_d01
| independent writes      = 2
| collective writes       = 638400
| independent writers     = 1
| aggregators             = 64
| stripe count            = 4
| stripe size             = 1048576
| system writes           = 82334
| stripe sized writes     = 82091
| aggregators active      = 177840,0,0,460560 (1,<=32,>32,64)
| total bytes for writes  = 82192 MiB = 80 GiB
| ave system write size   = 1046770
+--------------------------------------------------------+
```

# Application Evaluation, MPI-IO timers

- WRF example of MPI-IO timers (default locking)

```
+------------------------------------------------------------------------------+
| MPIIO write by phases, writers only, for wrfout_d01
|                                           min         max         ave
|                                       ----------  ----------  ----------
|   file write time               =       50.28       71.02       61.69
|   wait for coll                   =   27883170    38036293    33823494  5%
|   collective                      =     216151      311404      265380  0%
|   exchange/write                  =     432356      484632      463949  0%
|   data send                       =   52859942   101826753    75238921 12%
|   file write                      =  265243604   374646356   325433382 52%
|   other                           =  154855180   212509150   181886450 29%
|
|   data send BW (MiB/s)            =                             80.916
|   raw write BW (MiB/s)            =                           1332.366
|   net write BW (MiB/s)            =                            698.137
+------------------------------------------------------------------------------+
```

High file write percentage

# Application Evaluation, MPI-IO timers

- **WRF example of MPI-IO timers (LLA)**

```
+-------------------------------------------------------------------------+
| MPIIO write by phases, writers only, for wrfout_d01
|                                              min         max         ave
|                                       ----------  ----------  ----------
|   file write time                   =      2.90        6.31        4.72
|   wait for coll                     =  13154931    21812791    19040471   2%
|   collective                        =    866203     1252425     1064713   0%
|   exchange/write                    =   1748750     1934323     1870153   0%
|   data send                         =  69631810   145128634    97582694  11%
|   lock mode                         =    294934      423117      365912   0%
|   file write                        =  61251755   133053597    99598729  11%
|   other                             = 524346902   724441695   635362122  72%
|
|   data send BW (MiB/s)              =                            249.554
|   raw write BW (MiB/s)              =                          17413.726
|   net write BW (MiB/s)              =                           1982.863
+----------------------------------------------------------------------+
```

Equal data send and file write

# Application Quick Evaluation

- **Evaluating an existing application for use with LLA**
  1. Confirm collective MPI-IO writes and expected Lustre striping
  2. Using the data size and file system performance calculate if the improved throughput would be meaningful to overall application run time
  3. Confirm aggregator utilization as a basic indicator of I/O load using "`aggregators active`"
  4. Confirm MPI-IO write aggregators are currently spending a significant percentage of time in the "`file write`" phase

# Acknowledgements


- **Co-authors**
  - Patrick Farrell, Lustre client and server lockahead work
  - Bob Cernohous, MPI-IO LLA work (Cray MPT/ROMIO)
- **Paper contributors**
  - Bob Fiedler, Joe Glenski, Peter Johnsen, Norm Troullier, Richard Walsh
  - *Contributions listed in paper*

# Summary

- **Purpose**
  - Evaluate LLA for SSF performance in collective MPI-IO workloads
- **Results**
  - IOR performance shows SSF near FPP performance using LLA
  - WRF showed significantly decreased wall time using fewer storage resources by enabling LLA in Collective MPI-IO
  - Examples of using MPI-IO statistics and timers to evaluate the benefit of LLA for an existing application

# Legal Disclaimer

*Information in this document is provided in connection with Cray Inc. products. No license, express or implied, to any intellectual property rights is granted by this document.*

*Cray Inc. may make changes to specifications and product descriptions at any time, without notice.*

*All products, dates and figures specified are preliminary based on current expectations, and are subject to change without notice.*

*Cray hardware and software products may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.*

*Cray uses codenames internally to identify products that are in development and not yet publically announced for release. Customers and other third parties are not authorized by Cray Inc. to use codenames in advertising, promotion or marketing and any use of Cray Inc. internal codenames is at the sole risk of the user.*

*Performance tests and ratings are measured using specific systems and/or components and reflect the approximate performance of Cray Inc. products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance.*

*The following are trademarks of Cray Inc. and are registered in the United States and other countries: CRAY and design, SONEXION, and URIKA. The following are trademarks of Cray Inc.: APPRENTICE2, CHAPEL, CLUSTER CONNECT, CRAYPAT, CRAYPORT, ECOPHLEX, LIBSCI, NODEKARE, REVEAL, THREADSTORM. The following system family marks, and associated model number marks, are trademarks of Cray Inc.: CS, CX, XC, XE, XK, XMT, and XT. The registered trademark LINUX is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis. Other trademarks used in this document are the property of their respective owners.*

# Q&A

Michael Moore
mmoore@cray.com