Scaling MPT and Other Features

Mark Pagel

pags@cray.com

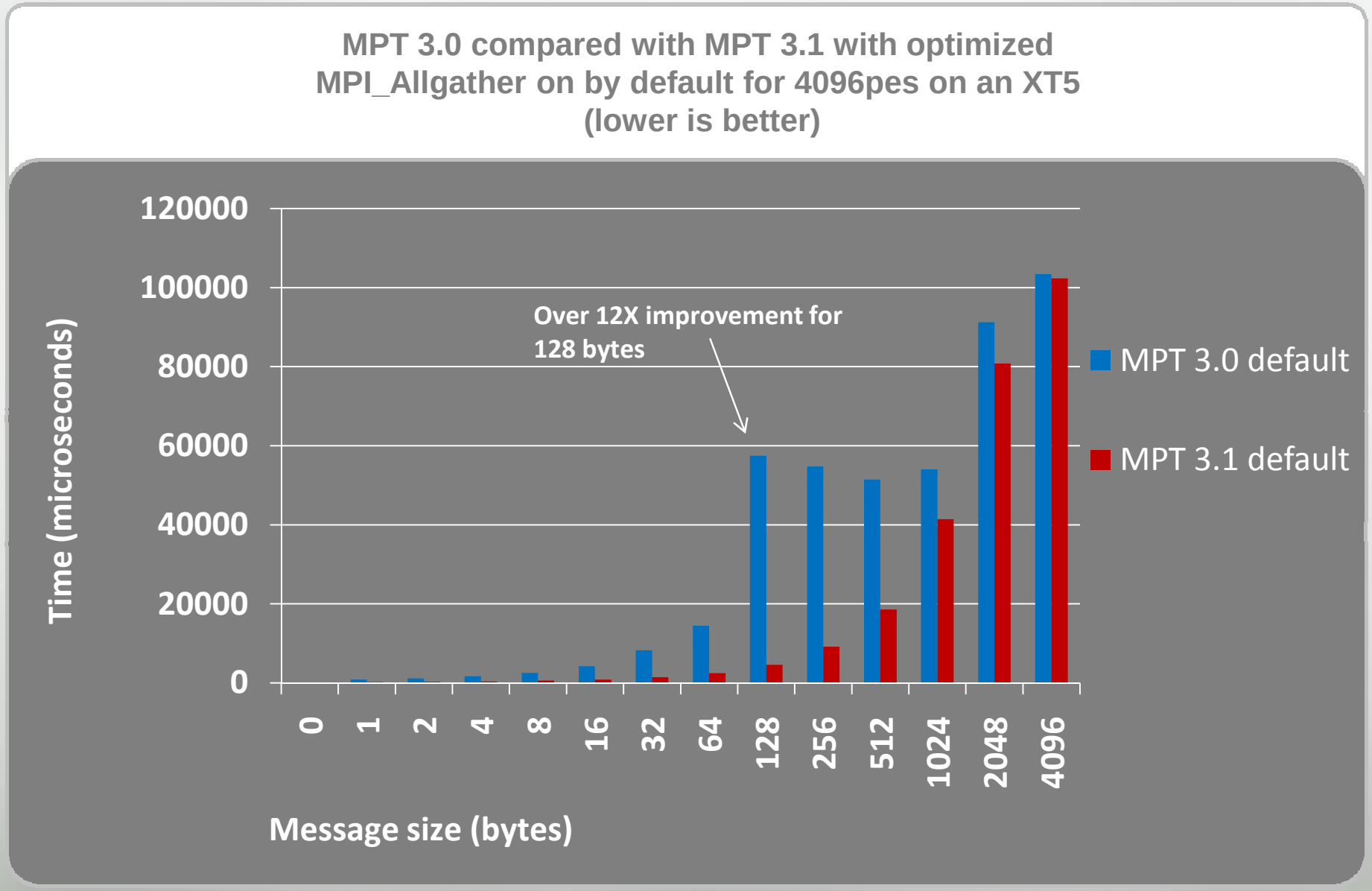
Outline

- New features in XT MPT 3.1 and MPT 3.2
  - Features as a result of scaling to 150K MPI ranks
  - MPI-IO Improvements(MPT 3.1 and MPT 3.2)
  - SMP aware collective improvements(MPT 3.2)
  - Misc Features (MPT 3.1)
- Future Releases

Features as a result of scaling to 150K MPI ranks (MPT 3.1)

- Support for over 256,000 MPI ranks
- Support for over 256,000 SHMEM PEs
- Automatically-tuned default values for MPI env vars
- Dynamic allocation of MPI internal message headers
- Improvements to start-up times when running at high process counts(40K cores or more)
- MPI_Allgather significant performance improvement

IMB MPI_Allgather performance

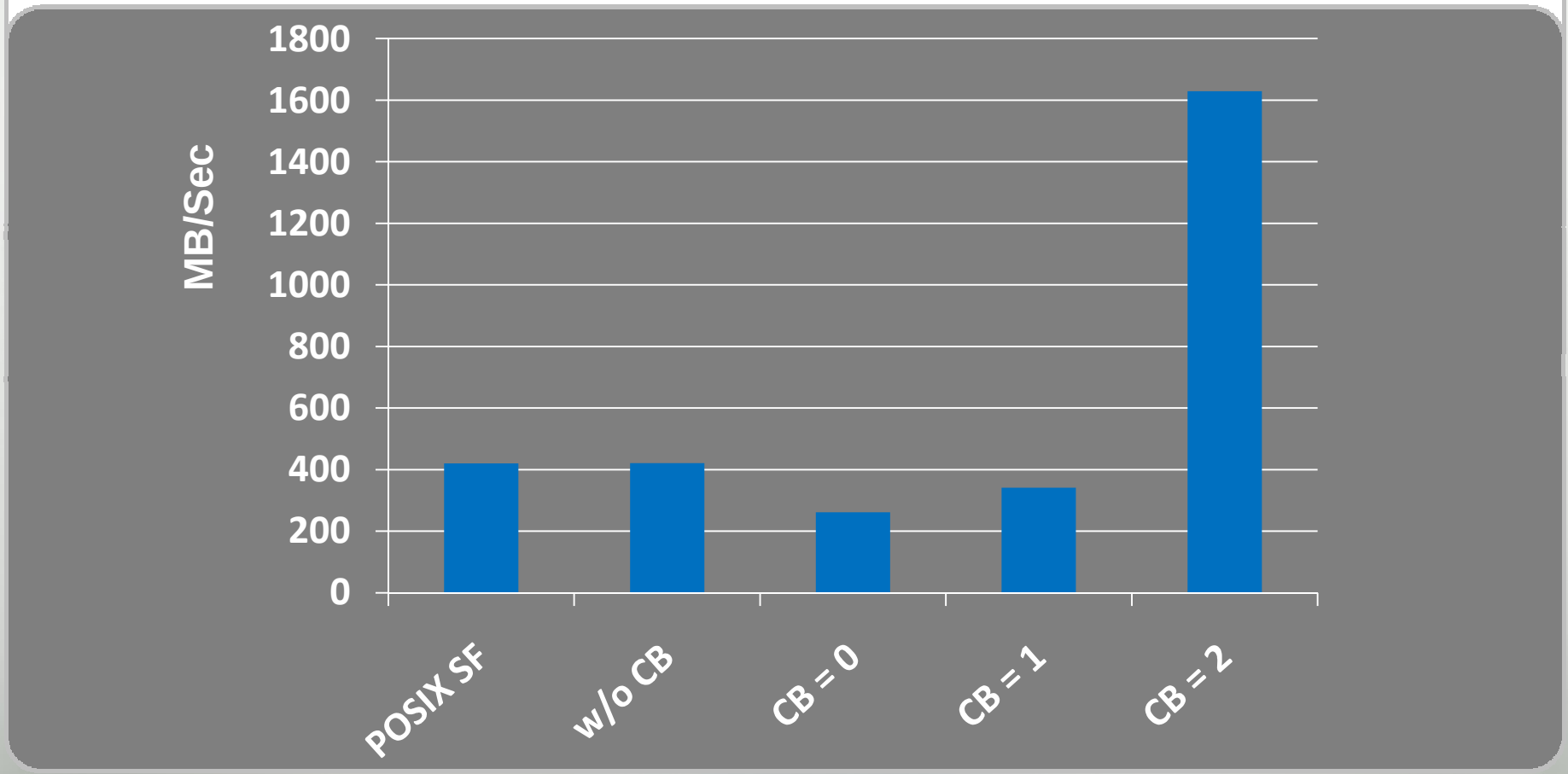


MPI-IO Improvements (MPT 3.1 and MPT 3.2)

- Wildcard matching for filenames in MPICH_MPIIO_HINTS (MPT 3.1)
- MPI-IO collective buffering alignment(MPT 3.1 and MPT 3.2)
  - This feature improves MPI-IO by aligning collective buffering file domains on Lustre boundaries.
  - The new algorithms take into account physical I/O boundaries and the size of the I/O requests. The intent is to improve performance by having the I/O requests of each collective buffering node (aggregator) start and end on physical I/O boundaries and to not have more than one aggregator reference for any given stripe on a single collective I/O call.
  - The new algorithms are enabled by setting the MPICH_MPIIO_CB_ALIGN env variable.
  - Additional enhancements in just released MPT 3.2

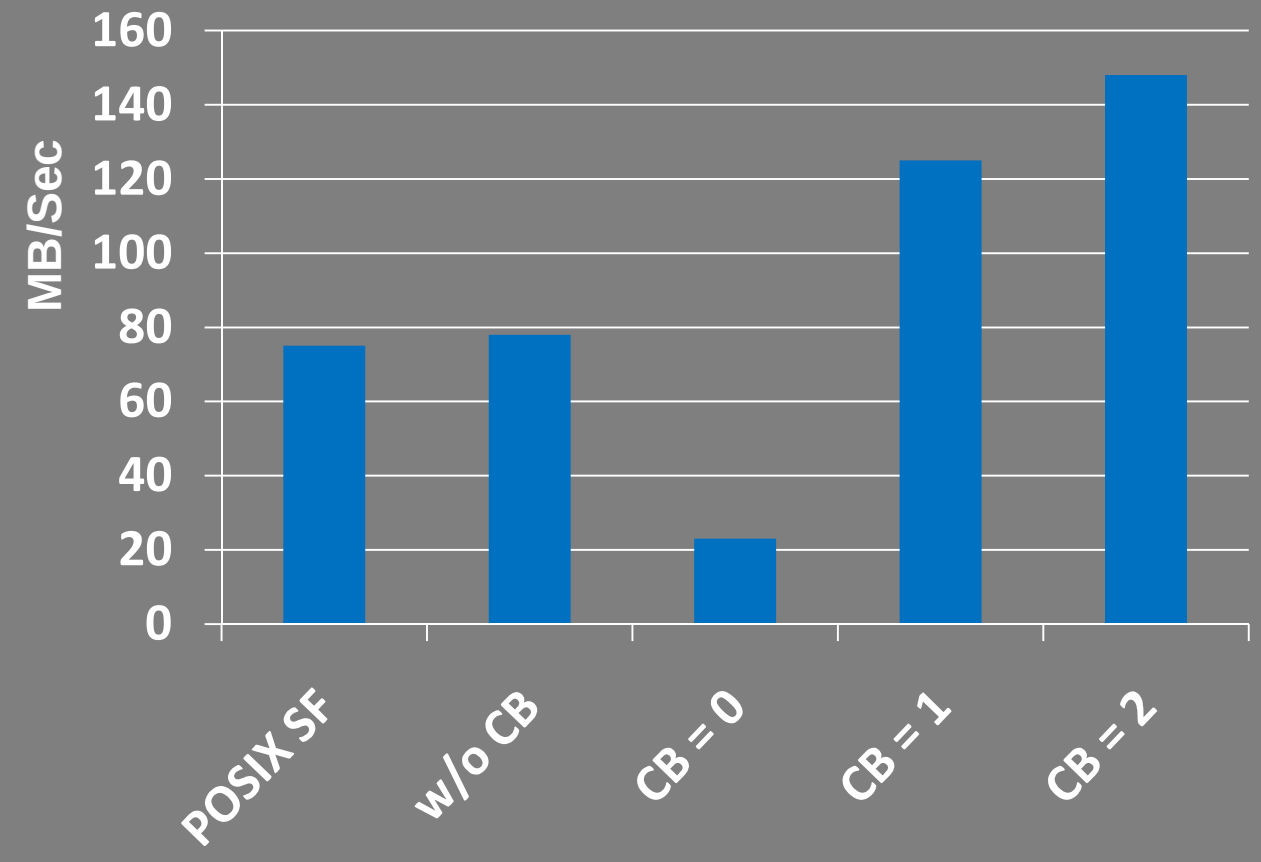
IOR benchmark 1,000,000 bytes

MPI-IO API , non-power-of-2 blocks and transfers, in this case blocks and transfers both of 1M bytes and a strided access pattern. Tested on an XT5 with 32 PEs, 8 cores/node, 16 stripes, 16 aggregators, 3220 segments, 96 GB file



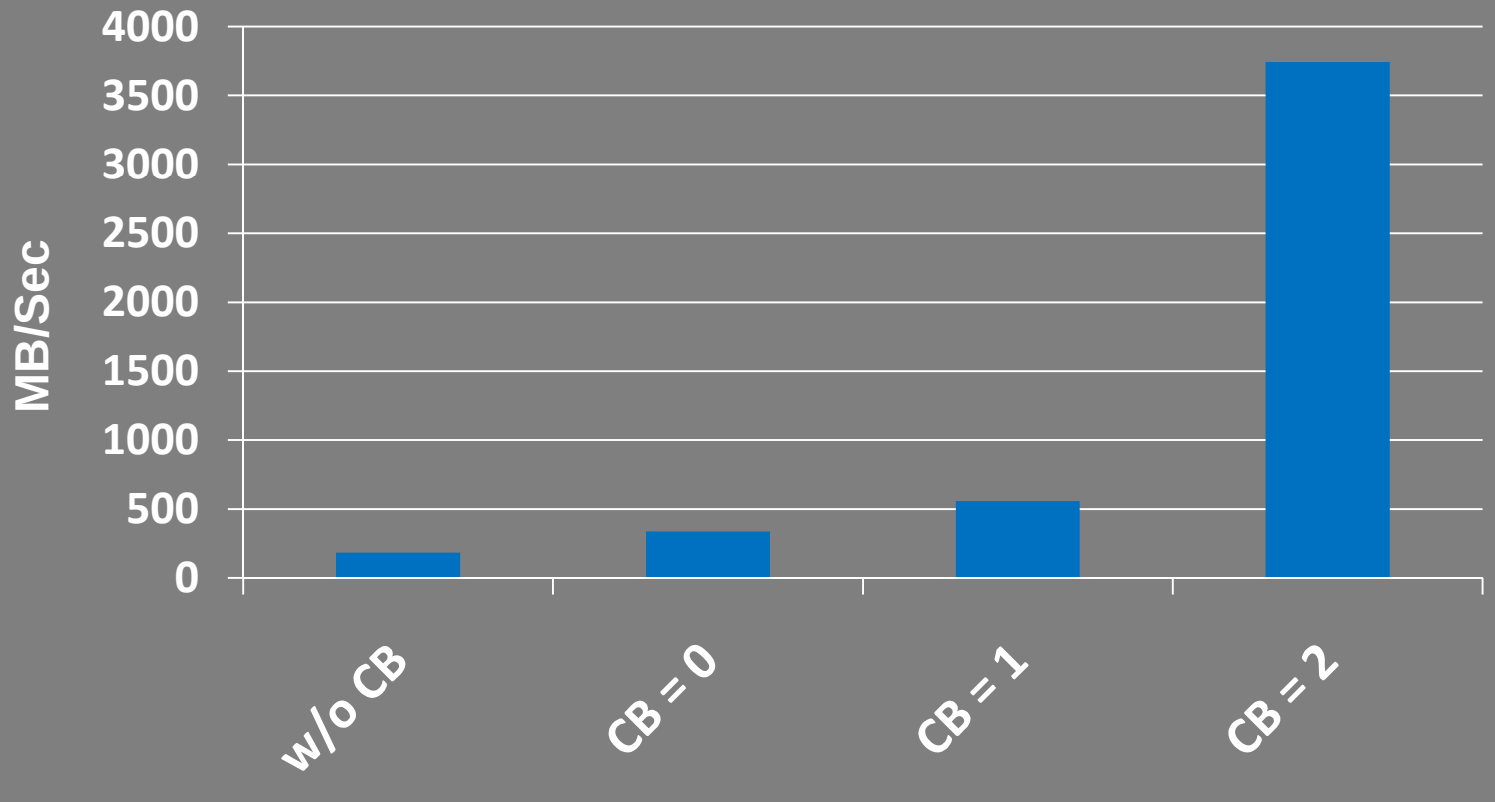
IOR benchmark 10,000 bytes

MPI-IO API , non-power-of-2 blocks and transfers, in this case blocks and transfers both of 10K bytes and a strided access pattern. Tested on an XT5 with 32 PEs, 8 cores/node, 16 stripes, 16 aggregators, 3220 segments, 96 GB file



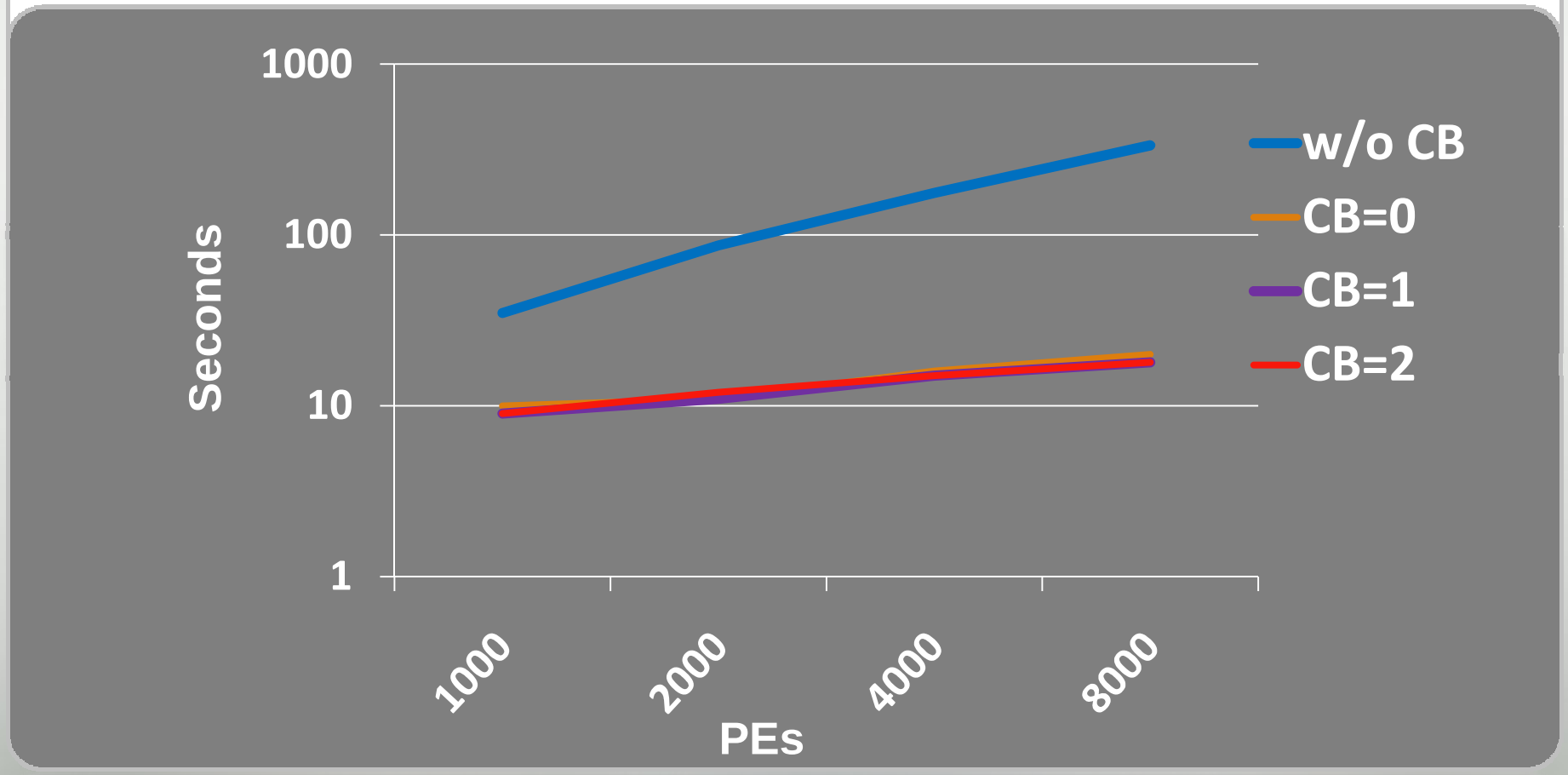
HYCOM MPI-2 I/O

On 5107 PEs, and by application design, a subset of the Pes(88), do the writes. With collective buffering, this is further reduced to 22 aggregators (cb_nodes) writing to 22 stripes. Tested on an XT5 with 5107 Pes, 8 cores/node



HDF5 format dump file from all PEs

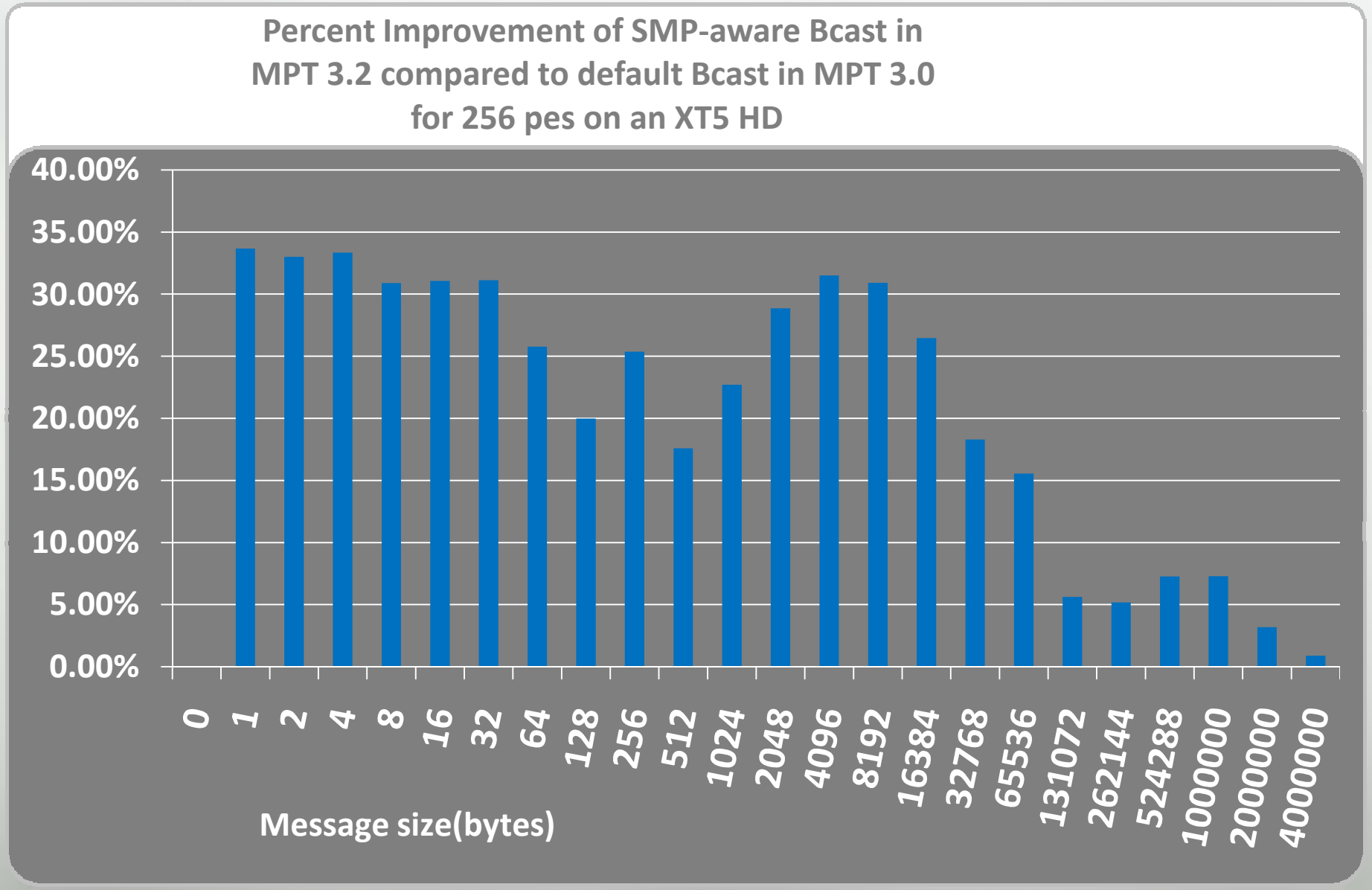
Total file size 6.4 GiB. Mesh of 64M bytes 32M elements, with work divided amongst all PEs. Original problem was very poor scaling. For example, without collective buffering, 8000 PEs take over 5 minutes to dump. Note that disabling data sieving was necessary. Tested on an XT5, 8 stripes, 8 cb_nodes



SMP aware collective improvements(MPT 3.2)

- MPI_Bcast has been optimized to be SMP aware
  - The performance improvement varies depending on message size and number of ranks but improvements of between 10% and 35% for messages below 128K bytes have been observed.
- MPI_Reduce has been optimized to be SMP aware
  - Performance improvements of over 3x for message sizes below 128K have been observed. A new environment variable MPICH_REDUCE_LARGE_MSG can be used to adjust the cutoff for when this optimization is enabled. See the man page for more info.

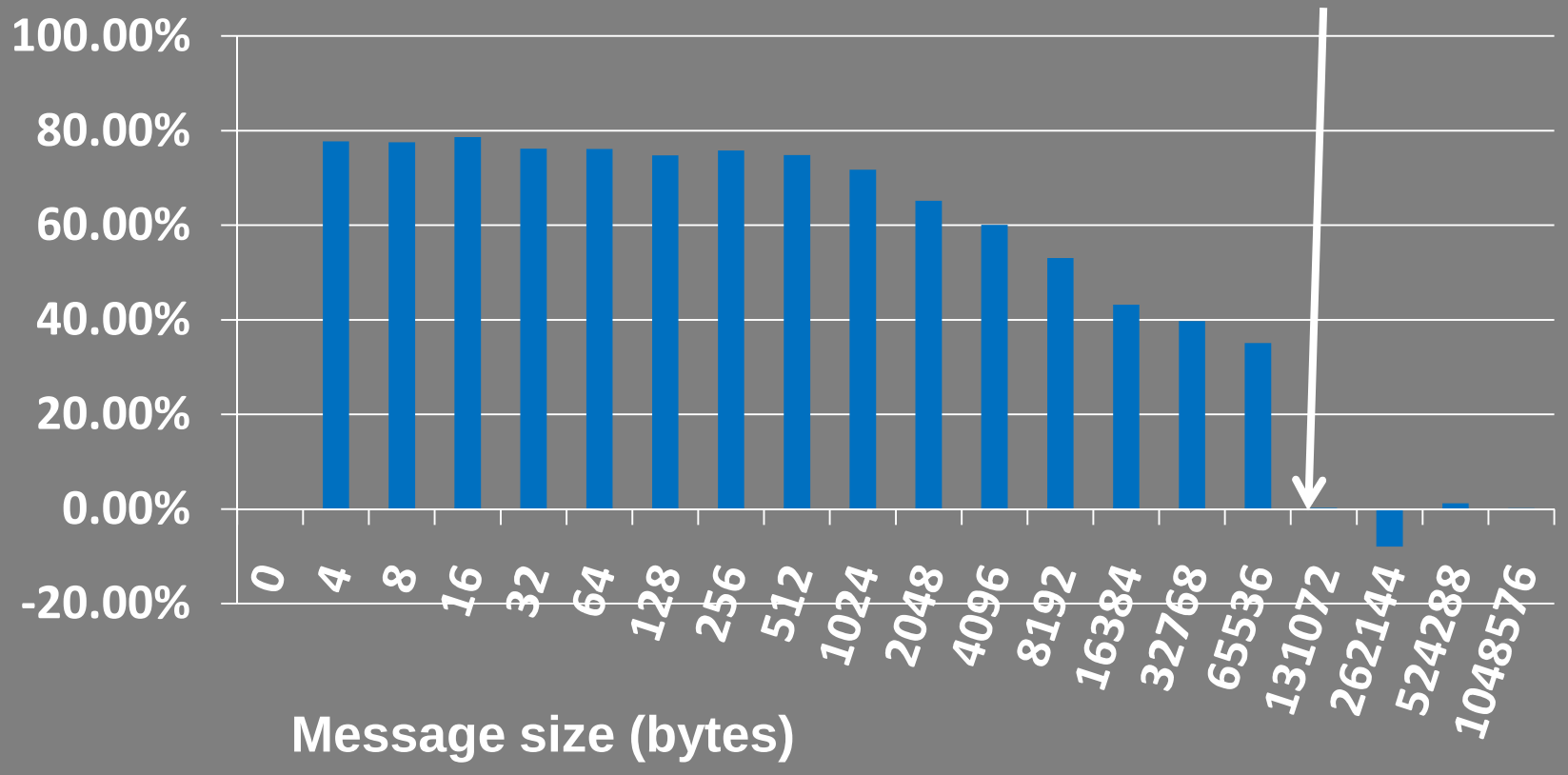
IMB MPI_Bcast performance



IMB MPI_Reduce performance

Percent Improvement of SMP-aware Reduce comparing default MPT 3.2 against default MPT 3.0 on XT5 with 256 PEs

For this chart we show what would happen if we didn't have the cutoff at 128K to switch back to the original algorithm. See mpi man page for more info on the MPICH_REDUCE_LARGE_MSG env variable.



Misc Features (MPT 3.1)

- Move from MPICH2 1.0.4p1 to MPICH2 1.0.6p1
- Cpu affinity support
- Support for the Cray Compiling Environment (CCE) 7.0
- MPI Barrier before collectives
- MPI Thread Safety
- MPI SMP device improvements for very large discontiguous messages
- Improvements have been made to the MPICH_COLL_OPT_OFF env variable(MPT 3.2)

Future Releases

- Bugfix updates every 4-8 weeks
- MPT 3.3 (scheduled for June 18, 2009)
  - Collective buffering enhancements from MPT 3.2 enabled as default
- MPT 4.0 (scheduled for Q4 2009)
  - Merge to ANL MPICH2 1.1
  - Support for the MPI 2.1 Standard
  - Additional MPI-IO Optimizations
  - Lustre ADIO device
  - Istanbul Support
  - Better performing MPI thread-safety (fine grain locking)

More Info

- Man pages
  - intro_mpi
  - Intro_shmem
  - aprun
- MPI-IO white paper

  (ftp://ftp.cray.com/pub/pe/download/MPI-IO_White_Paper.pdf)
- MPI Standard documentation

  (http://www.mpi-forum.org/docs/docs.html)
- MPICH2 implementation information

  (http://www-unix.mcs.anl.gov/mpi/mpich2)

Questions?


CRAY
THE SUPERCOMPUTER COMPANY

Additional slides for xtra detail…

New features in XT MPI 3.1 (released 12/11/08)

- Move from MPICH2 1.0.4p1 to MPICH2 1.0.6p1
  - Performance improvements for derived datatypes and MPI_Gather
  - MPI_Comm_create now works for intercommunicators.
  - Many other bug fixes, memory leak fixes and code cleanup.
  - Fixes for regressions in MPICH2 1.0.6p1 that were fixed in MPICH2 1.0.7.
- Cpu affinity support
  - This allows MPI processes to be pinned to a specific CPU or set of CPUs, as directed by the user via the new aprun affinity and placement options. Affinity support is provided for both MPI and MPI/OpenMP hybrid applications.

XT MPI 3.1 Features (continued)

- Support for over 64,000 MPI ranks
  - New limit for how high MPI jobs can scale on XT systems. The new limit is 256,000 MPI ranks.
- Support for over 32,000 SHMEM PEs
  - New limit for how high SHMEM jobs can scale on XT systems. The new limit is 256,000 SHMEM PEs.
  - In order to support higher scaling, changes were made to the SHMEM header files that require a recompile when using this new version. The new library will detect this incompatibility and issue a FATAL error message telling you to recompile with the new headers.

XT MPI 3.1 Features (continued)

- Automatically-tuned default values for MPICH environment variables
  - Higher scaling of MPT jobs with fewer tweaks to environment variables.
  - User can override by setting the environment variable.
  - The env variables affected are: MPICH_MAX_SHORT_MSG_SIZE, MPICH_PTL_OTHER_EVENTS MPICH_PTL_UNEX_EVENTS, MPICH_UNEX_BUFFER_SIZE
- Dynamic allocation of MPI internal message headers
  - Apps no longer abort if it runs out of headers, and require MPICH_MSGS_PER_PROC environment variable to be increased. Now MPI dynamically allocates more message headers in quantities of MPICH_MSGS_PER_PROC

XT MPI 3.1 Features (continued)

- Improvements to start-up times when running at high process counts(40K cores or more)
  - This change significantly reduces our MPI_Init startup time on very large jobs. For example for a 86,000 PE job, start-up time went from 280 seconds down to 128 seconds.
- MPI_Allgather significant performance improvement
  - New MPI_Allgather collective routine which scales well for small data sizes. The default is to use the new algorithm for any MPI_Allgather calls with 2048 bytes of data or less. Can be changed by setting a new env varariable called MPICH_ALLGATHER_VSHORT_MSG.
  - Some MPI functions use allgather internally and will now be significantly faster. For example MPI_Comm_split.
  - Initial results show improvements of around 2X around 16 cores to over 100X above 20K cores.

XT MPI 3.1 Features (continued)

- Wildcard matching for filenames in MPICH_MPIIO_HINTS
  - Allows easier specification of hints for multiple files that are opened with MPI_File_open in the program. The filename pattern matching follows standard shell pattern matching rules for meta-characters ?, \\, [], and *.
- Support for the Cray Compiling Environment (CCE) 7.0 compiler
  - Allows the x86 ABI compatible mode of the Cray Compiling Environment (CCE) 7.0 to be compatible with the Fortran MPI bindings for that compiler.

XT MPI 3.1 Features (continued)

- MPI Barrier before collectives
  - In some situations an MPI_Barrier inserted before a collective may improve performance due to load imbalance. This feature adds support for a new environment variable MPICH_COLL_SYNC which will cause a MPI_Barrier call to be inserted before all collectives or only certain collectives.
  - To enable this feature for all MPI collectives, set MPICH_COLL_SYNC to 1 or a comma separated list of collectives. See man page for more info.

XT MPI 3.1 Features (continued)

- MPI-IO collective buffering alignment
  - This feature improves MPI-IO by aligning collective buffering file domains on Lustre boundaries.
  - The new algorithms take into account physical I/O boundaries and the size of the I/O requests. The intent is to improve performance by having the I/O requests of each collective buffering node (aggregator) start and end on physical I/O boundaries and to not have more than one aggregator reference for any given stripe on a single collective I/O call.
  - The new algorithms are enabled by setting the MPICH_MPIIO_CB_ALIGN env variable.
  - Additional enhancements in just released MPT 3.2

XT MPI 3.1 Features (continued)

- MPI Thread Safety
  - The MPI Thread Safety feature provides a high-performance implementation of thread-safety levels MPI_THREAD_SINGLE, MPI_THREAD_FUNNELED, and MPI_THREAD_SERIALIZE in the main MPI library.
  - The MPI_THREAD_MULTIPLE thread-safety level support is in a separate "mpich_threadm" library and is not a high-performance implementation.
  - Use "-lmpich_threadm" when linking to MPI_THREAD_MULTIPLE routines.
  - Set MPICH_MAX_THREAD_SAFETY environment variable to the desired level.

XT MPI 3.1 Features (continued)

- MPI SMP device improvements for very large discontiguous messages
  - This feature enables a new algorithm for the on-node SMP device to process large discontiguous messages. The new algorithm allows the use of our on-node Portals-assisted call that is used in our MPT 3.0 single-copy feature rather than buffering the data into very small chunks as was currently being done.
  - Some applications have seen as much as a 3X speedup with discontiguous messages in excess of 4M bytes.

MPT 3.1.x update releases

- Initial MPT 3.1 – released on 12/11/08
- MPT 3.1.1 – released on 2/19/09
  - 744991 turning on Argonne memory debugging causes RMA tests to fail
  - 745025 Memory leakage in a communication routine when using self defined MPI-datatypes.
  - 745198 MPI communication in SMP mode seems to block single sided comms
  - 746416 data type mismatch in MPI-IO sometimes causes I/O stats to be displayed
  - 747035 IMB-EXT fails intermittently with MPT 3
  - 747183 Application CTH reports being killed by OOM Killer when compiled with MPT 3.0 or 3.1

MPT 3.1.x update releases

- MPT 3.1.2 – released on 3/19/09
  - 747752 Unsupported datatype error from MPI library
  - 747708 Seg fault in program using RMA mpi_get and mpi_send/mpi_recv
- Current outstanding issues:
  - 744363 GAMESS claims to have run out of memory and be killed by oom when it should not
- Update releases every 4-8 weeks

MPT 3.2 Features (Released 4/16/2009)

- MPI_Bcast has been optimized to be SMP aware and this optimization is enabled by default. The performance improvement varies depending on message size and number of ranks but improvements of between 10% and 35% for messages below 128K bytes have been observed.

- MPI_Reduce has been optimized to be SMP aware and this optimization is enabled by default. The SMP aware algorithm performs significantly better than the default algorithm for most message sizes. Performance improvements of over 3x for message sizes below 128K have been observed. A new environment variable MPICH_REDUCE_LARGE_MSG can be used to adjust the cutoff for when this optimization is enabled. See the man page for more info.

MPT 3.2 (April 16th 2009)

- Improvements have been made to the MPICH_COLL_OPT_OFF environment variable by allowing a finer-grain switch to enable/disable the optimized collectives. The user may now:
  - Enable all of the optimized collectives (this is the default)
  - Disable all the opt collectives (export MPICH_COLL_OPT_OFF=0)
  - Disable a selected set of the optimized collectives by providing a comma-separated list of the collective names

    e.g. export MPICH_COLL_OPT_OFF=MPI_Allreduce,MPI_Bcast

    Note: If a user chooses to disable any Cray-optimized collective, they will get the standard MPICH2 algorithm.

- Bugs fixed:
  - 748815 - segfault in MPIDI_CRAY_search_posted

MPT 3.2 (April 16th 2009)

- MPI-IO performance improvements for collective buffering on MPI collective writes
  - This optimization is enabled by setting the MPI-IO hint romio_cb_write to "enable" and setting the environment variable MPICH_MPIIO_CB_ALIGN to 2.
  - Other values of this environment variable are 0 and 1, where 0 is for the original algorithm in MPT 3.0 and earlier and 1 is for the algorithm introduced in MPT 3.1.
  - The MPICH_MPIIO_CB_ALIGN section of the "mpi" man page gives more details. If you are not already using collective buffering, read the MPICH_MPIIO_HINTS section for more information.


CRAY
THE SUPERCOMPUTER COMPANY