Optimizing Cray MPI and SHMEM Software Stacks for Cray-XC Supercomputers based on Intel KNL Processors

May 12, 2016

Krishna Kandalla, Peter Mendygral, Nick Radcliffe, Bob Cernohous, David Knaak, Kim McMahon, Mark Pagel

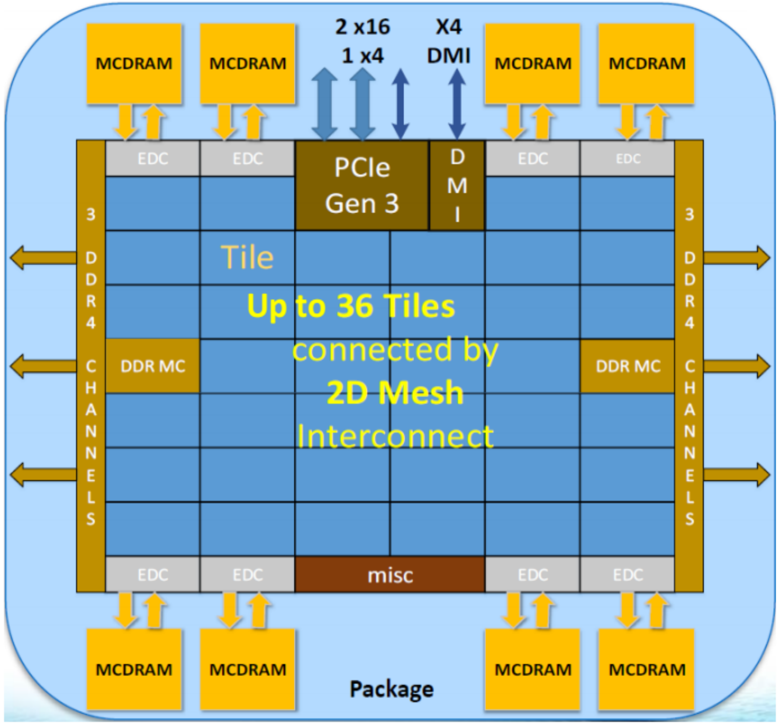
(**kkandalla**, pjm, nradcliff,bcernohous,knaak,kmcmahon,pags)@cray.com

Agenda

- Introduction & Motivation
- Problem Statement
- Design and Methodology
- Experimental Evaluation
- Summary & Contributions
- Q&A

Introduction & Motivation

- Intel KNL offers at least 64 cores per node, 2 TF double precision performance per chip
- Different from Xeon – wider vectors, slower cores, slower scalar processing
- MPI is ubiquitous - allows applications to scale beyond tens of thousands of nodes
- Easy way to hit the ground running on a KNL – pack each KNL node with many MPI processes
- Packing a KNL with MPI processes leads to resource constraints (memory footprint ..)



Intel KNL Architecture

Introduction & Motivation

- **Hybrid (MPI + OpenMP) models allow fewer MPI processes per node, while utilizing all cores to accelerate compute**
- **“Bottom-Up” development approach is very common.**

  May not always offer best performance
- **“Top-Down” SPMD model is more appealing for KNL**

  Increases the scope of code executed by OpenMP, allows for better load balancing and overall compute scaling on KNL. (John Levesque talk at CUG)
  - Allows multiple threads to call MPI concurrently.
  - In this model, performance is limited by the level of support offered by MPI for multi-threaded communication
  - MPI implementations must offer “Thread-Hot” communication capabilities to improve communication performance for highly threaded use cases on KNL

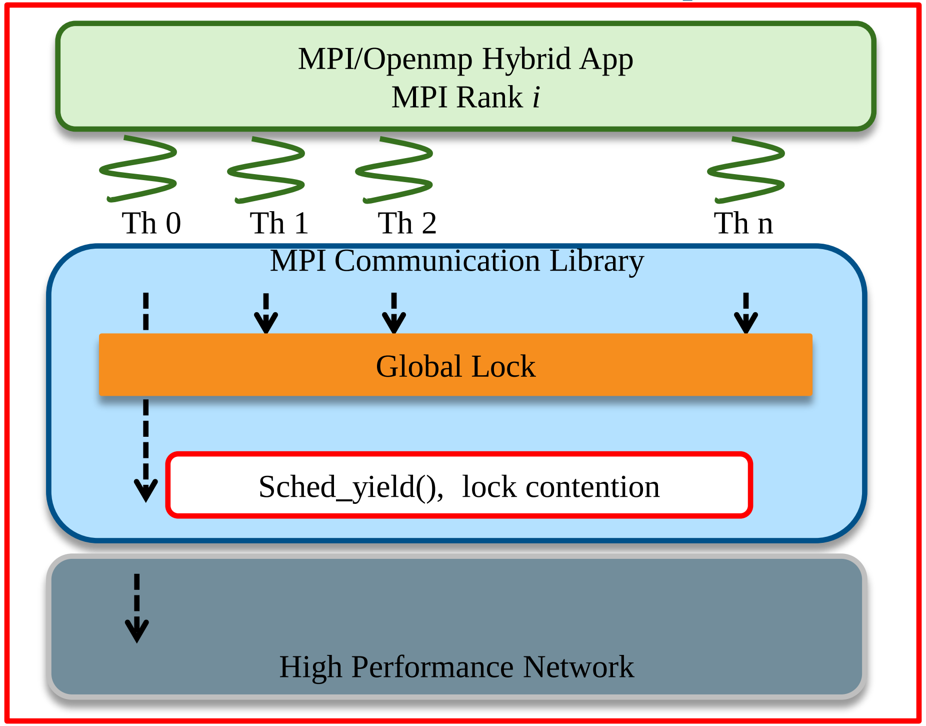
Introduction & Motivation

- KNL offers MCDRAM: On package memory
- MCDRAM can be configured as “flat” or “cache”
- KNL also offers NUMA modes: A2A, Hemi, Quad, SNC2, and SNC4
- System software stacks (MPI & SHMEM), compilers and parallel applications need to evolve to best utilize this technology.
- Software support necessary to manage specialized memory (such as huge page backed memory) on MCDRAM.

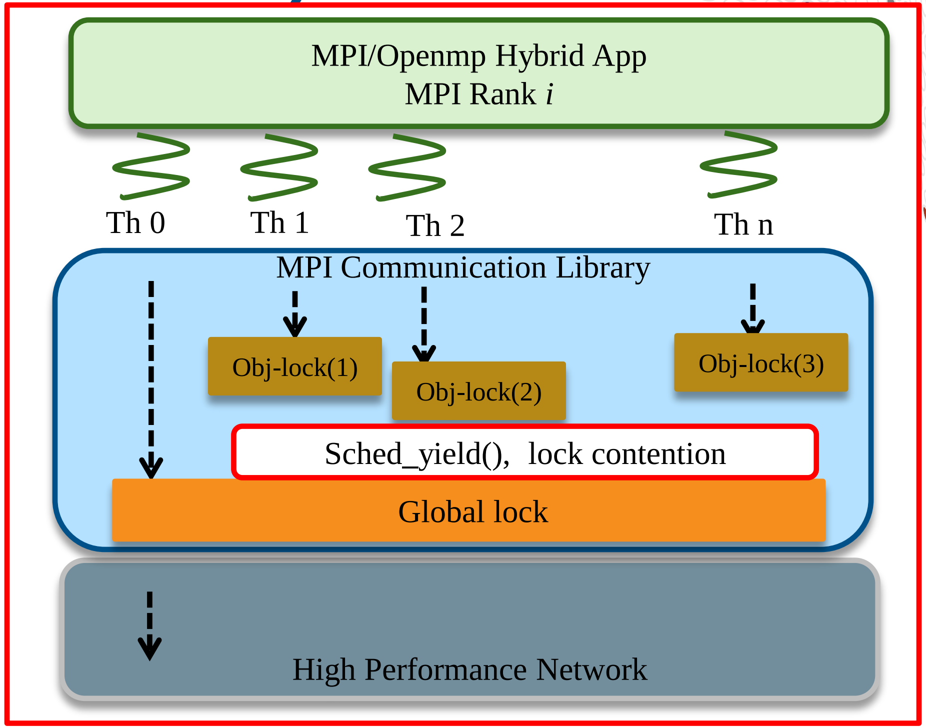
Agenda

- Introduction & Motivation
- Problem Statement
  - Designing Thread Hot MPI
  - Managing specialized memory on KNL
- Design and Methodology
- Experimental Evaluation
- Summary & Contributions
- Q&A

Multi-Threaded MPI (State-Of-The-Art)



Global lock
(default in Cray MPI)



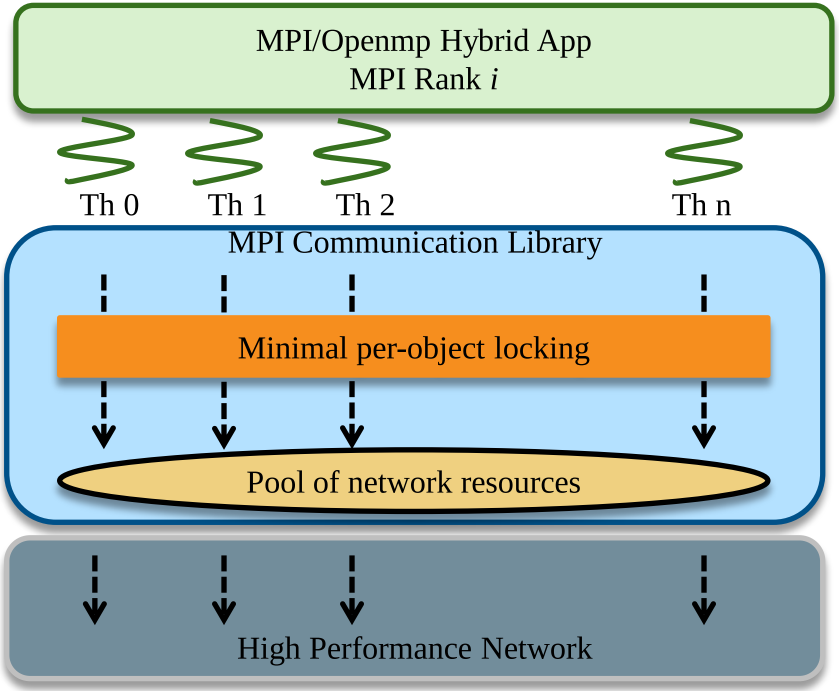
Per-Object Locks
(Alt. impl. in CrayMPI, “–craympich-mt”)

Multi-Threaded MPI Optimizations

- MPI implementations relying on a single global lock cannot offer high performance multi-threaded communication
- "Thread-Hot" MPI communication is required to improve application performance of Top-Down Hybrid applications
- Cray MPI offers an alternate per-object implementation that relies on fine-grained locking mechanisms for MPI pt2pt operations

  This implementation still uses the global lock around specific layers
- ***Can new solutions be designed to allocate a set of software/hardware resources and dynamically manage them across threads to offer high performance communication with minimal locking overheads?***
- ***Can MPI implementations be designed to support Thread-Hot communication for a range of MPI operations: pt2pt, RMA and collectives?***

Optimized Multi-Threaded MPI (Design Choices)



Proposed Thread-Hot Design

Enqueue/Dequeue Design

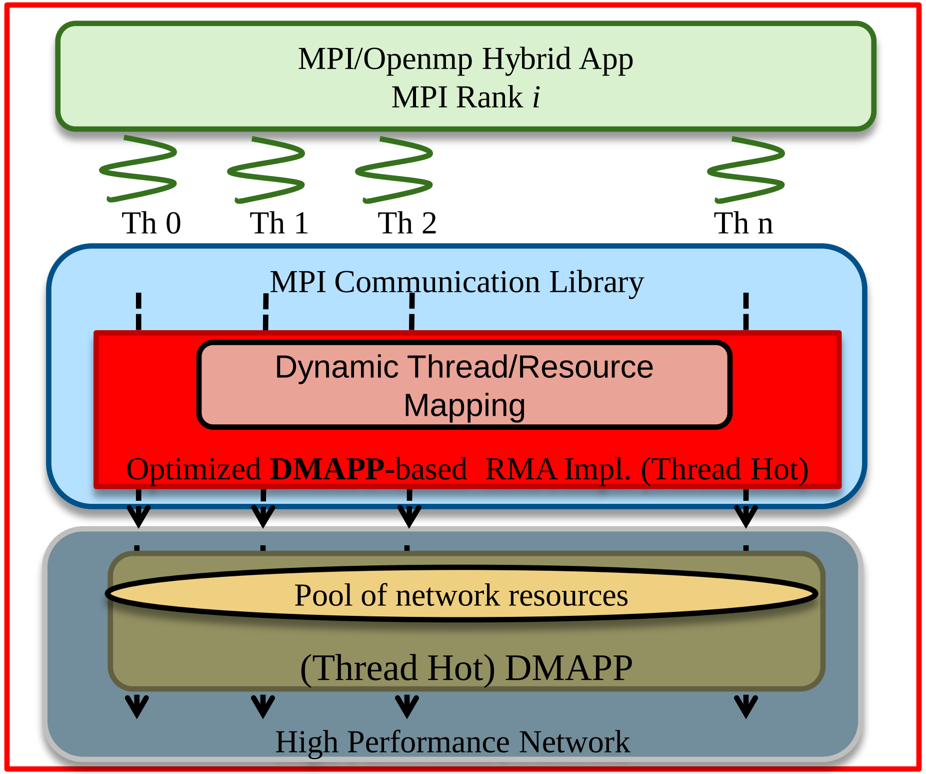
KNL High Bandwidth Memory (MCDRAM)

- Several ways to allocate memory on MCDRAM for KNL
  - CCE or Intel Compiler directives
  - memkind API (hbw_malloc)
  - numactl
  - Explicit mmap/mbind OS calls (non-trivial for end users)
- But getting hugepage memory on MCDRAM is difficult
  - Using hugepages is recommended to achieve good performance on XC
  - memkind does NOT pay attention to the craype-hugepages modules
    - even if craype-hugepage module is loaded, memkind uses 4KB pages!
  - memkind API has some hugepage options
    - Only 2M and 1GB page sizes are supported in the API
    - ..but 1GB pages are not supported on CLE
  - CCE/Intel compiler directives can't request MCDRAM hugepages currently
- Can MPI and SHMEM implementations offer new solutions to allow hugepage memory on MCDRAM?
  - Should work for Quad/SNC2/SNC4 modes
  - Should work with MCDRAM partially or fully configured in “flat” mode

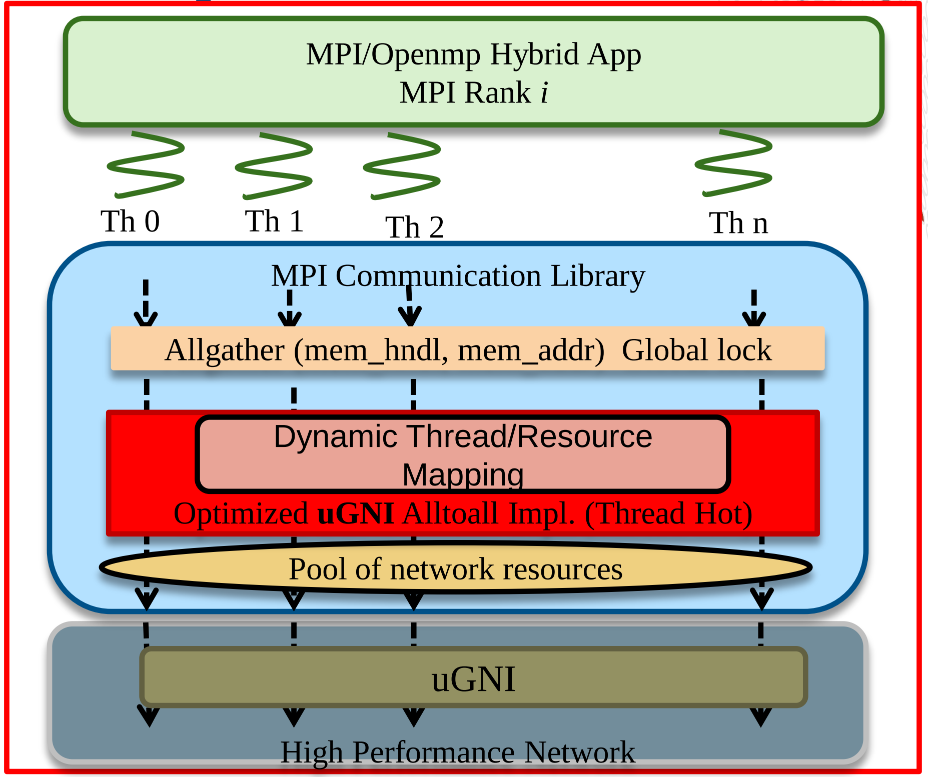
Agenda

- Introduction & Motivation
- Problem Statement
- Design and Methodology
  - Thread Hot MPI Communication in Cray MPI
  - Designing WOMBAT for high performance and scalability
  - MPI & SHMEM support for KNL on Cray XC
- Experimental Evaluation
- Summary & Contributions
- Q&A

Thread Hot Communication in Cray MPI



Thread Hot MPI-3 RMA



Thread Hot MPI_Alltoall

Thread Hot Communication in Cray MPI

- Design Objectives
  - Contention Free progress and completion
  - High bandwidth and high message rate
  - Independent progress – One thread flushes outstanding traffic, other threads make uninterrupted progress
  - Dynamic mapping between threads and network resources
  - Locks needed only if the number of threads exceed the number of network resources

- MPI-3 RMA
  - Epoch calls (Win_complete, Win_fence) are thread-safe, but not intended to be thread hot
  - Multiple threads calling Win_start and Win_complete will open multiple epochs; instead of accelerating one
  - All other RMA calls (including request-based operations) are thread hot
  - Multiple threads doing Passive Synchronization operations likely to perform best:

- MPI_Alltoall
  - Multiple threads can issue, progress and complete Alltoall operations concurrently. Each thread has a separate MPI_Comm handle.
  - The Allgather exchange (mem address, hndls) is protected by the big lock (room for optimization)

Agenda

- Introduction & Motivation
- Problem Statement
- Design and Methodology
  - Thread Hot MPI Communication in Cray MPI
  - Designing WOMBAT for high performance and scalability
  - MPI & SHMEM support for KNL on Cray XC
- Experimental Evaluation
- Summary & Contributions
- Q&A

Hybrid MPI/OpenMP Applications: Design Alternatives

Option A: ***(Top Down)***

```
! Move OpenMP near the top of the call stack

!#OMP PARALLEL
DO WHILE (t .LT. tend)

   !#OMP DO SCHEDULE(GUIDED)
   DO work = 1, work_end

      CALL update_work()

      ! All threads drive MPI

   END DO

END DO
```

Option B: ***(Bottom Up)***

```
! Keep OpenMP within a "compute" loop

DO WHILE (t .LT. tend)
   DO work = 1, work_end
      CALL update_work()
      ! MPI driven by single thread
   END DO
END DO

SUBROUTINE update_work()
   !$OMP PARALLEL DO
SCHEDULE(STATIC)
   DO i = 1, nx
   …do work…
   END DO
END SUBROUTINE
```

Designing WOMBAT for high performance and scalability

- WOMBAT is a shock capturing magneto-hydrodynamic (MHD) code
- Studies a number of astrophysical phenomena -- outflows from super massive black holes, evolution of galactic super-bubbles, and MHD turbulence in intra-cluster medium in galaxy clusters
- WOMBAT supports scientific goals of studying MHD turbulence at very high resolution using a combination of static and adaptive mesh-refinement strategies
- Developed through a collaboration between Cray Inc. and the University of Minnesota Institute of Astrophysics
- This work addresses the challenges in optimizing WOMBAT on modern processors such as KNL. (beyond 10^5 cores)
- Time consuming solvers involve nearest neighbor sub-volume communication

Designing WOMBAT for high performance and scalability

- Load balancing critical – work must be explicitly moved form overloaded ranks to less loaded ranks
- Fewer MPI ranks with many OpenMP threads can reduce frequency of load balancing
- MPI-3 RMA used to implement near-neighbor communication (instead of MPI Pt2Pt)
- "Top-Down" MPI/OpenMP approach. OpenMP threads will call RMA operations concurrently and independently
- If MPI can offer high performance multi-threaded RMA communication, significant opportunity for optimizing the performance and scalability of WOMBAT
- *A significant fraction of the code must be multi-threaded, and MPI must eliminate need for thread-synchronization to optimize performance.*

Agenda

- Introduction & Motivation
- Problem Statement
- Design and Methodology
  - Thread Hot MPI Communication in Cray MPI
  - Designing WOMBAT for high performance and scalability
  - Cray MPI & Cray SHMEM support for KNL on Cray XC
- Experimental Evaluation
- Summary & Contributions
- Q&A

Cray MPI support for MCDRAM on KNL

- **Cray MPI will offer hugepage support for MCDRAM on KNL**
  - Must use: MPI_Alloc_mem() or MPI_Win_Allocate()
  - Dependencies: memkind and NUMA libraries
- **Preliminary release will expose feature via env variables**
  - Users select: Affinity, Policy and PageSize
  - MPICH_ALLOC_MEM_AFFINITY = DDR or MCDRAM
    - DDR = allocate memory on DDR (default)
    - MCDRAM = allocate memory on MDCRAM
  - MPICH_ALLOC_MEM_POLICY = M/ P/ I
    - M = Mandatory: fatal error if allocation fails
    - P = Preferred: fall back to using DDR memory (default)
    - I = Interleaved: Set memory affinity to interleave across MCDRAM NUMA
  - MPICH_ALLOC_MEM_PG_SZ
    - 4K, 2M, 4M, 8M, 16M, 32M, 64M, 128M, 256M, 512M (default 4K)
- **Follow-on release will offer Info Key Support for MPI_Alloc_me m and MPI_Win_allocate**
  - Allows user to specify characteristics via Info parameter on each call

Cray SHMEM support for MCDRAM on KNL

- **SHMEM support for MCDRAM on KNL**
  - Cray working with Intel to define a common API for SHMEM
  - Requires use of Intel's memkind library, and libnuma
  - Control memory placement via env variables
  - New env variable: SMA_SYMMETRIC_PARTITION#
  - User specifies: Size, Kind, Policy and PgSize
    - size=<any valid size based on available memory>
    - kind=D|Default|F|Fastmem (D=DDR, F=MCDRAM)
    - policy=M|Mandatory|P|Preferred|I|Interleaved
    - pgsize=<Supported pagesizes>
  - Can set up multiple partitions with different characteristics
  - Original shmalloc calls use memory from Partition1
  - Two new SHMEM API calls
    - `void *shmem_kind_malloc(size, partition_id)`
    - `void *shmem_kind_aligned_alloc(alignment, size, partition_id)`
  - Allocates 2 GB of MCDRAM memory using 2MB hugepages and aborts it the allocation fails

SMA_SYMMETRIC_PARTITION1=size=2G:kind=F:policy=M:pgsize=2M

Agenda

- Introduction & Motivation
- Problem Statement
- Design and Methodology
- Experimental Evaluation
  - Thread Hot MPI Optimizations
  - Wombat Scaling
- Summary & Contributions
- Q&A

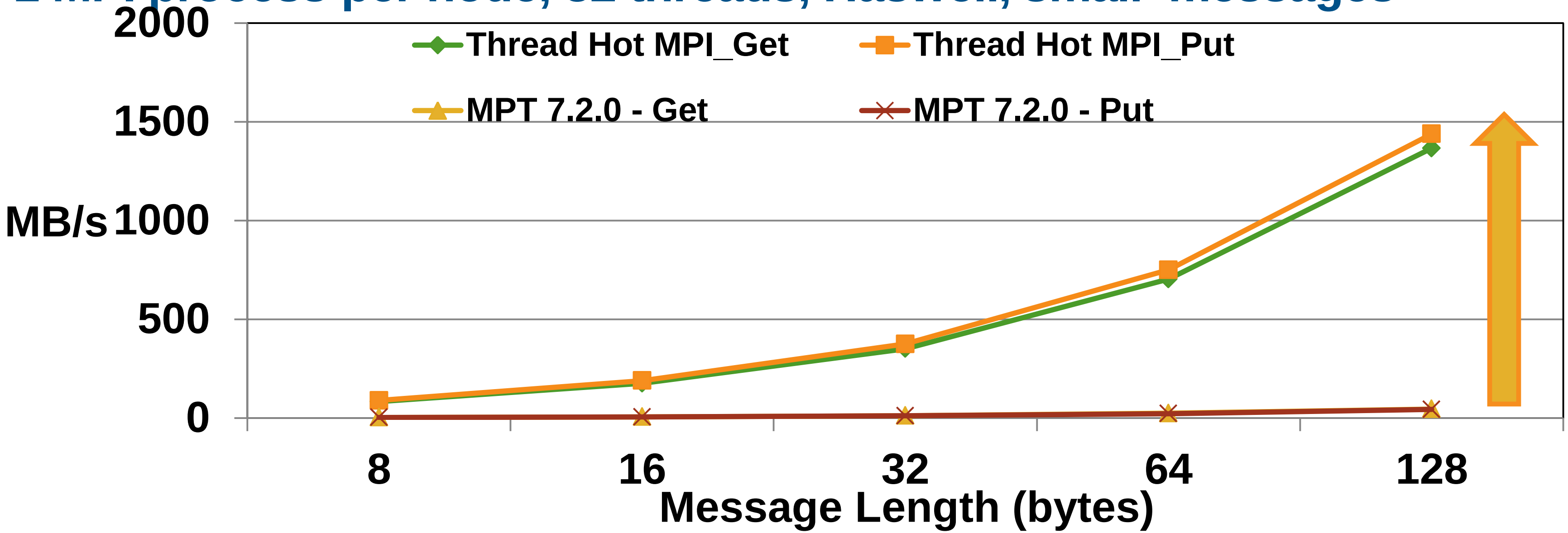
Experimental Setup

- Cray XC systems with Intel Haswell and Broadwell
- *Modified* OSU Micro Benchmarks (OMB) to study multi-threaded MPI Communication performance
  - RMA: osu_put_latency.c, osu_get_latency.c
    osu_put_bw.c, osu_get_bw.c
  - Collective: osu_alltoall.c

- Proposed designs are also showing significant improvements on Cray XC with KNL

MPI-3 RMA Communication Bandwidth

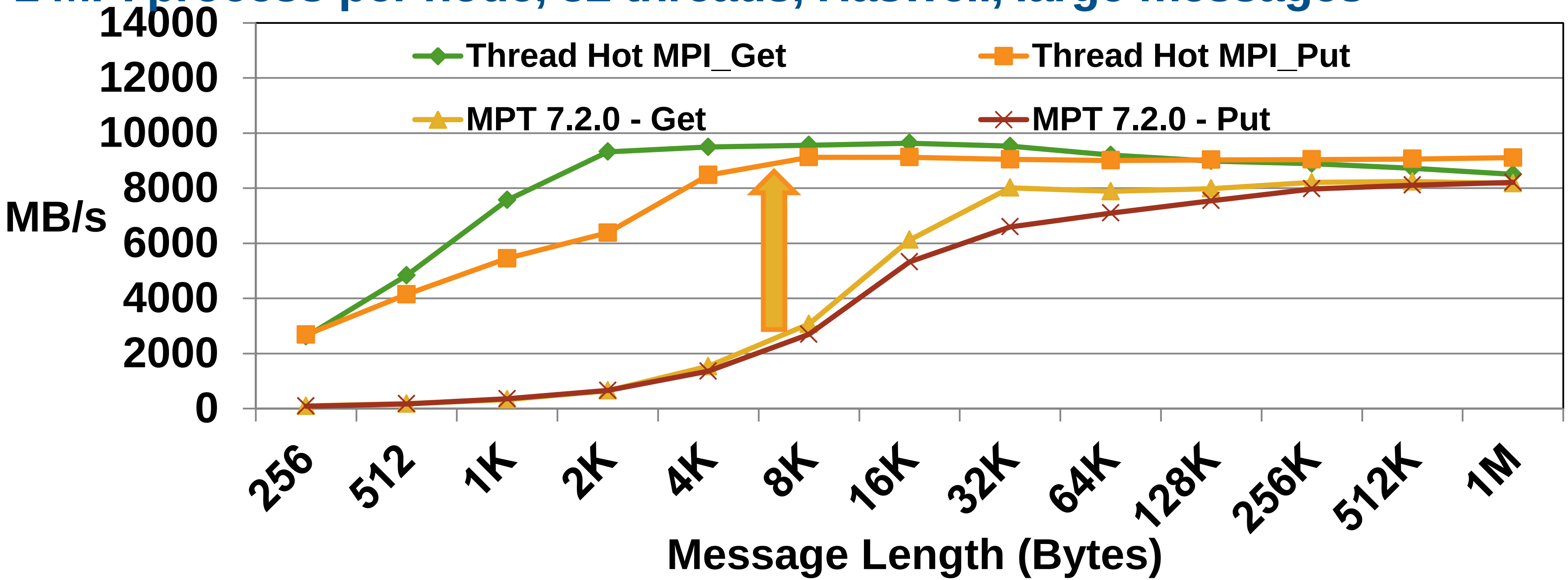
1 MPI process per node, 32 threads, Haswell, small messages



- Thread Hot Cray MPI significantly outperforms the default (global-lock) implementation with the multi-threaded RMA benchmark for small payloads

MPI-3 RMA Communication Bandwidth

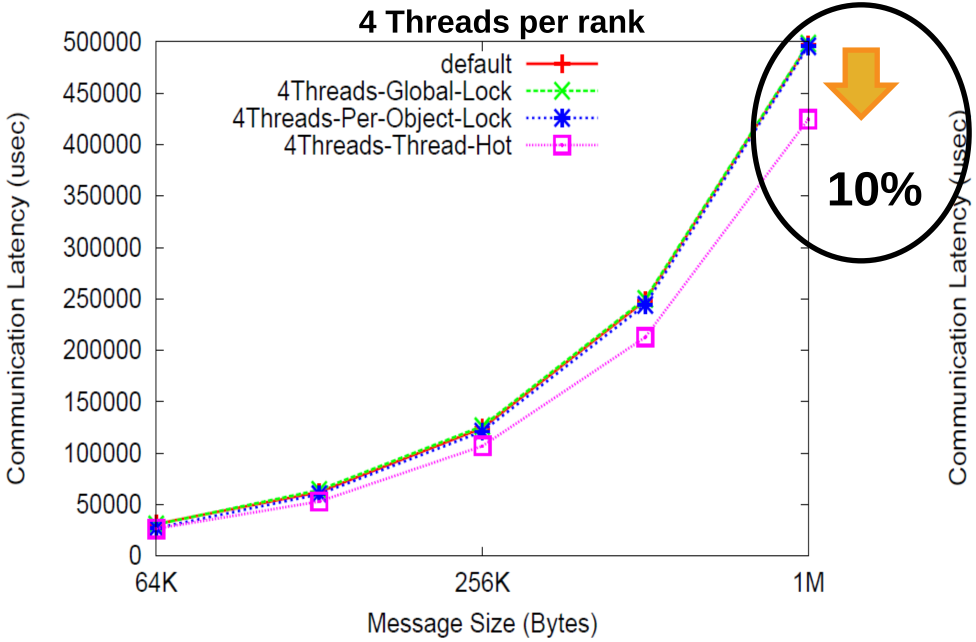
1 MPI process per node, 32 threads, Haswell, large messages

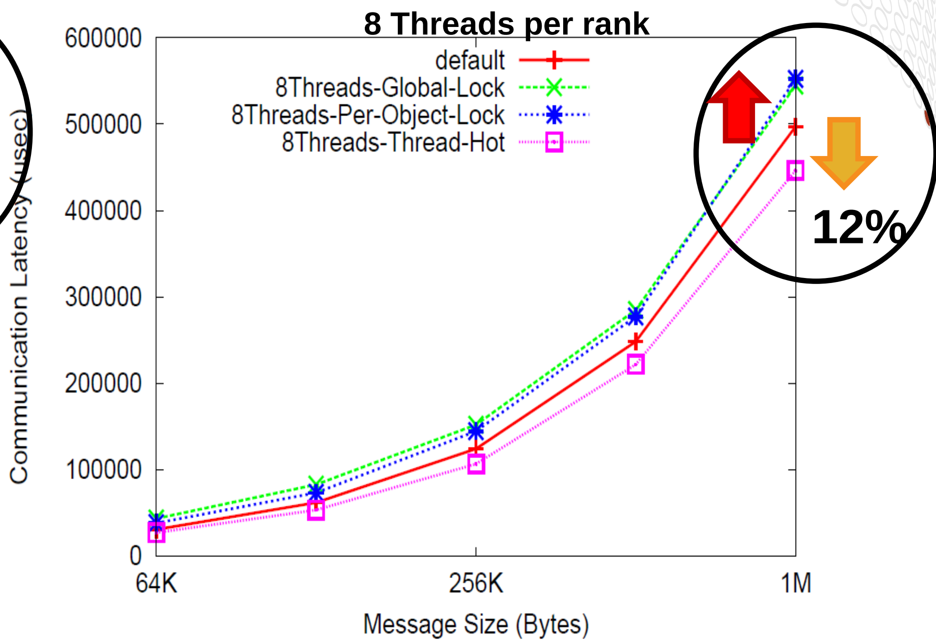


- Thread Hot Cray MPI outperforms the default (global-lock) implementation with the multi-threaded RMA benchmark by about 4X for small and medium sized payloads

COMPUTE | STORE | ANALYZE

MPI_Alltoall Performance
128 Nodes, 512 MPI Processes



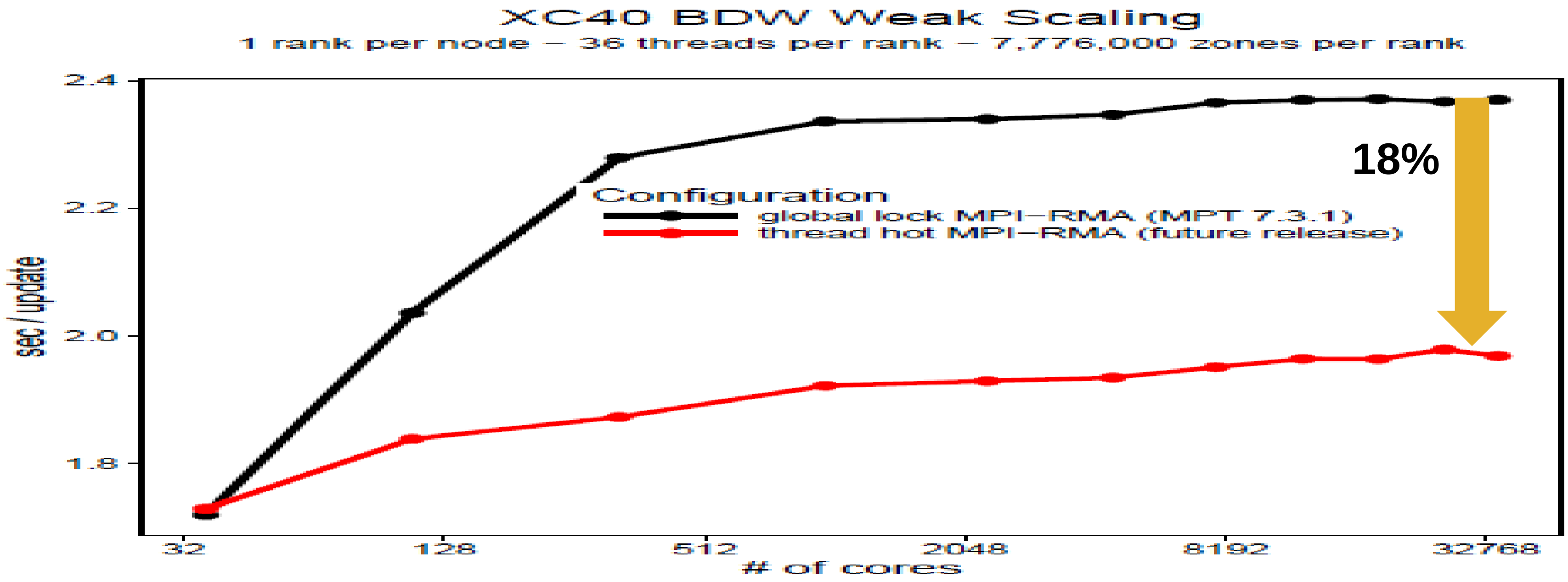


- With increasing number of threads per rank, performance degradation observed with global and per-object locks
- Proposed Thread Hot implementation improves multi-threaded communication latency by more than 10%

Agenda

- Introduction & Motivation
- Problem Statement
- Design and Methodology
- Experimental Evaluation
  - Thread Hot MPI Optimizations
  - Wombat Scaling
- Summary & Contributions
- Q&A

WOMBAT Weak Scaling Results



34,848 Intel Broadwell cores - from MPI only to wide OpenMP 36 threads per rank, 1 rank per node
Thread Hot RMA offers more than 18% reduction in time required to perform an "update" in WOMBAT

Summary and Contributions

- New solutions in Cray MPI to offer Thread-Hot capabilities on Intel Xeon and Intel KNL architectures
- Design and development details of Wombat, a high performance astrophysics application that relies on multi-threaded MPI-3 RMA implementation in Cray MPI
- Enhancements in Cray MPI and Cray SHMEM software stacks to enable users best utilize the MCDRAM technology on KNL

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, THREADSTORM, REVEAL. 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

Krishna Kandalla
kkandalla@cray.com