Argonne NATIONAL LABORATORY

MPICH: A High-Performance Open-Source MPI Implementation

Birds-of-a-Feather Session

Supercomputing 2015

Austin, TX

November 17, 2015

U.S. DEPARTMENT OF ENERGY

Schedule

[5:30] Welcome and Overview

[5:35] MPICH Big Picture Plans

[6:00] Cray Update

[6:05] Intel Update

[6:10] Lenovo Update

[6:15] Mellanox Update

[6:20] Microsoft Update

[6:25] RIKEN Update

[6:30] Parastation Update

[6:35] Tianhe Update

[6:40] FG MPI Update

[6:45] Q&A and Wrapup

MPICH:
Current Status and Upcoming Releases

Pavan Balaji

Computer Scientist

Group Lead, Programming Models and Runtime systems

Argonne National Laboratory

MPICH turns 23

The MPICH Project

- MPICH and its derivatives are the world's most widely used MPI implementations
  - Supports all versions of the MPI standard including the recent MPI-3
- Funded by DOE for 23 years (turned 23 this month)
- Has been a key influencer in the adoption of MPI
- Award winning project
  - DOE R&D100 award in 2005

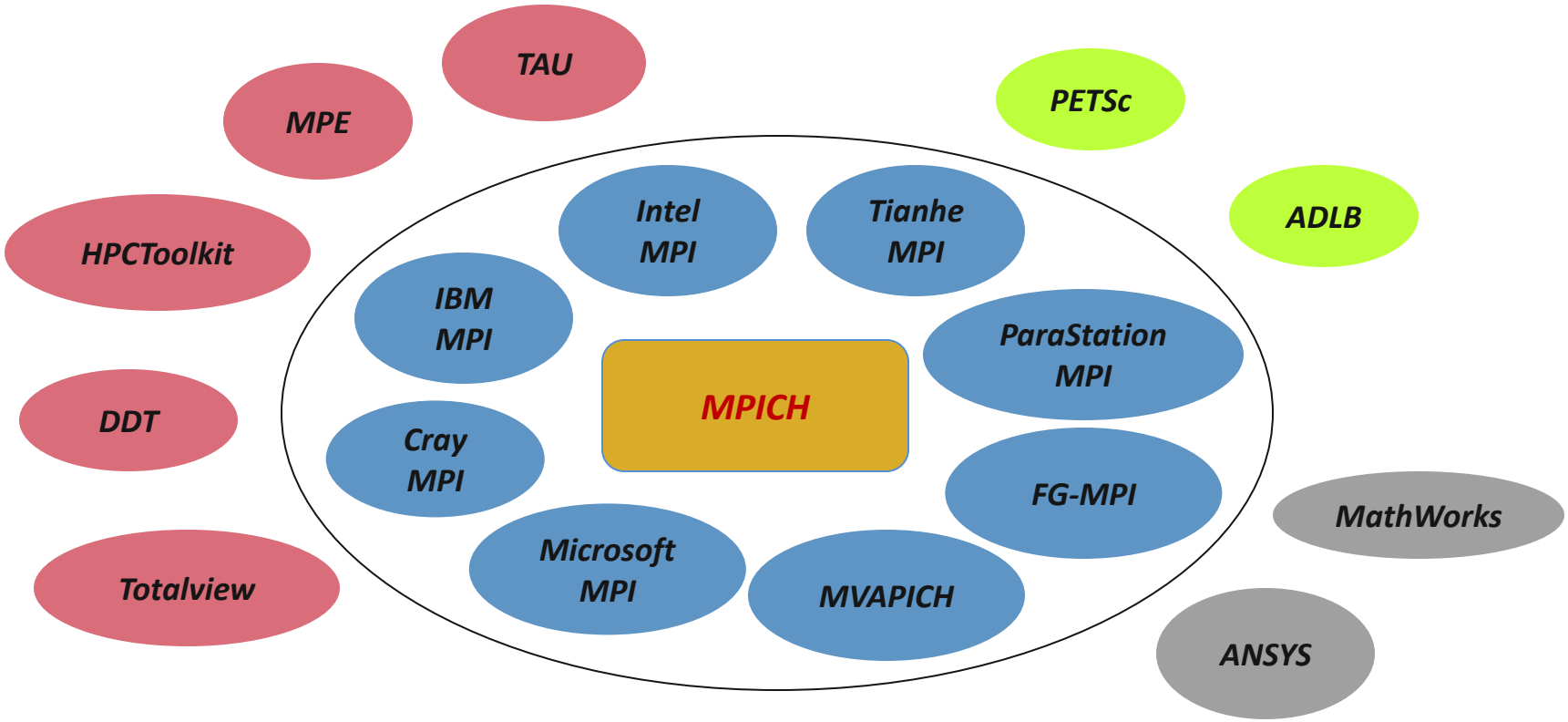
MPICH and it derivatives in the Top 10

1. ***Tianhe-2 (China): TH-MPI***
2. ***Titan (US): Cray MPI***
3. ***Sequoia (US): IBM PE MPI***
4. K Computer (Japan): Fujitsu MPI
5. ***Mira (US): IBM PE MPI***
6. ***Trinity (US): Cray MPI***
7. ***Piz Daint (Germany): Cray MPI***
8. ***Hazel Hen (Germany): Cray MPI***
9. ***Shaheen II (Saudi Arabia): Cray MPI***
10. ***Stampede (US): Intel MPI and MVAPICH***

MPICH and its derivatives power 9 of the top 10 supercomputers (Nov. 2015 Top500 rankings)

MPICH: Goals and Philosophy

- MPICH continues to aim to be the preferred MPI implementations on the top machines in the world
- Our philosophy is to create an "MPICH Ecosystem"



MPI-3.1 Implementation Status: MPICH Derivatives

	MPICH	MVAPICH	Cray MPI	Tianhe MPI	Intel MPI	IBM BG/Q MPI[1]	IBM PE MPICH[2]	MS MPI
NBC	✔	✔	✔	✔	✔	✔	✔	(*)
Nbrhood collectives	✔	✔	✔	✔	✔	✔	✔	
RMA	✔	✔	✔	✔	✔	✔	✔	
Shared memory	✔	✔	✔	✔	✔	✔	✔	✔
Tools Interface	✔	✔	✔	✔	✔	✔	✔	*
Comm-creat group	✔	✔	✔	✔	✔	✔	✔	
F08 Bindings	✔	✔	✔	✔		✔		
New Datatypes	✔	✔	✔	✔	✔	✔	✔	✔
Large Counts	✔	✔	✔	✔	✔	✔	✔	✔
Matched Probe	✔	✔	✔	✔	✔	✔	✔	✔
NBC I/O	✔	Q1'16	Q4'15					

Release dates are estimates and are subject to change at any time.

Empty cells indicate no *publicly announced* plan to implement/support that feature.

Platform-specific restrictions might apply for all supported features

[1] Open Source but unsupported [2] No MPI_T variables exposed * Under development (*) Partly done

MPICH ABI Compatibility Initiative

- Runtime compatibility for MPI implementations
  - Explicit goal of maintaining ABI compatibility between multiple MPICH derivatives
  - Initial collaborators include:
    - MPICH (since v3.1, 2013)
    - IBM PE MPI (since v1.4, 2014)
    - Intel MPI Library (since v5.0, 2014)
    - Cray MPT (starting v7.0, 2014)
  - More details at http://www.mpich.org/abi
- Open initiative: other MPI implementations are welcome to join

MPICH 3.2 Feature Update

Full MPI-3.1 functionality

Support for a number of network APIs (MXM, HCOLL, OFI, LLC, Portals 4)

Revamped RMA infrastructure (highly scalable and low-overhead)

(Very preliminary) Support for User-level Fault Mitigation

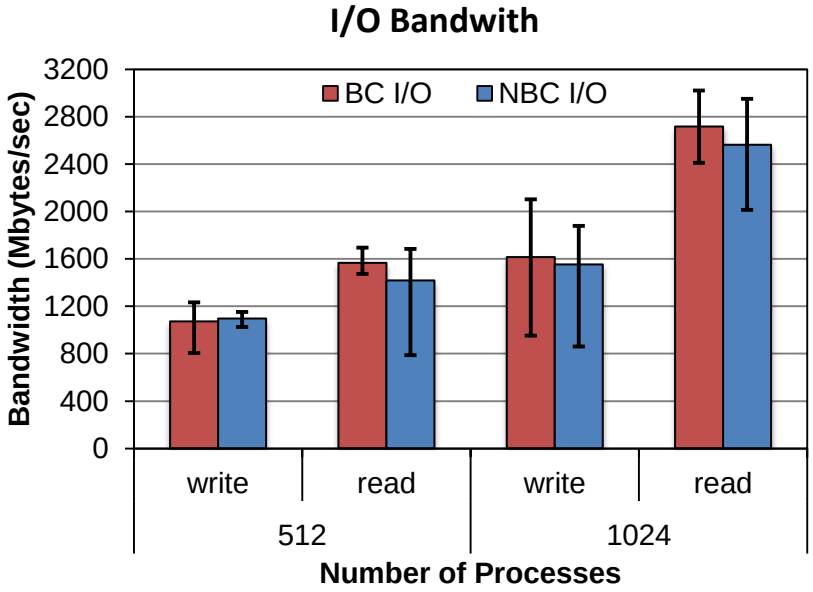
MPICH-3.2

- MPICH-3.2 is the latest major release series of MPICH
  - Released mpich-3.2 on Nov. 11th, 2015
- Primary focus areas for mpich-3.2
  - Support for MPI-3.1 functionality (nonblocking collective I/O and others)
  - Fortran 2008 bindings
  - Support for the Mellanox MXM interface (thanks to Mellanox)
  - Support for the Mellanox HCOLL interface (thanks to Mellanox)
  - Support for the LLC interface for IB and Tofu (thanks to RIKEN)
  - Support for the OFI interface (thanks to Intel)
  - Improvements to MPICH/Portals 4
  - MPI-4 Fault Tolerance (ULFM)
  - Major improvements to the RMA infrastructure

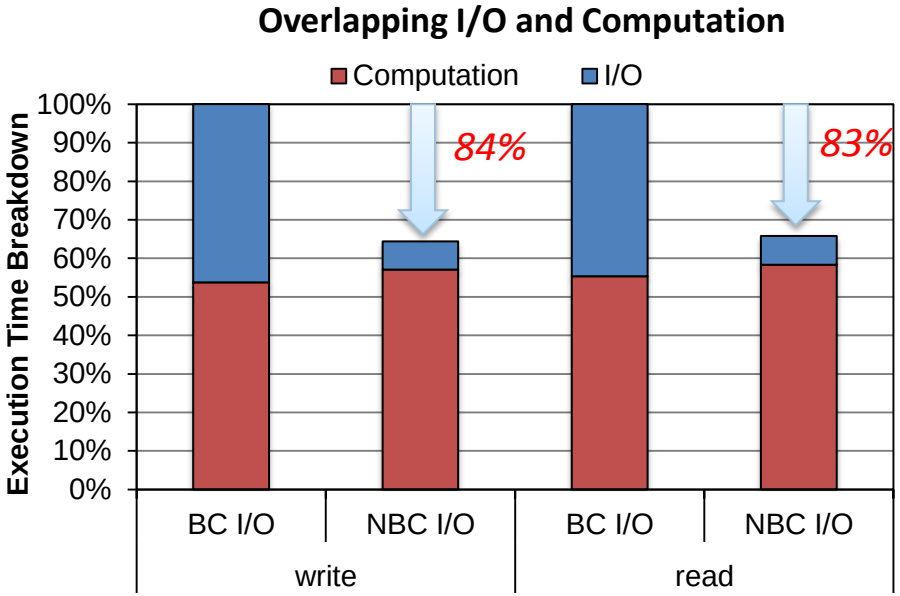
MPI-3.1 Nonblocking Collective (NBC) I/O

Sangmin Seo
Assistant Computer Scientist

- Included in MPI 3.1 standard: Immediate versions of blocking colle
  - **MPI_File_read_all**(MPI_File fh, void *buf, int count, MPI_Datatype dataty MPI_Status *status)
  - **MPI_File_iread_all**(MPI_File fh, void *buf, int count, MPI_Datatype datatype, MPI_Request *request)
  - Same for **MPI_File_iread_at_all**, **MPI_File_iwrite_all**, and **MPI_File_iwrite_at_all**
- Supported in MPICH 3.2:
  - Implemented in ROMIO using the extended generalized request



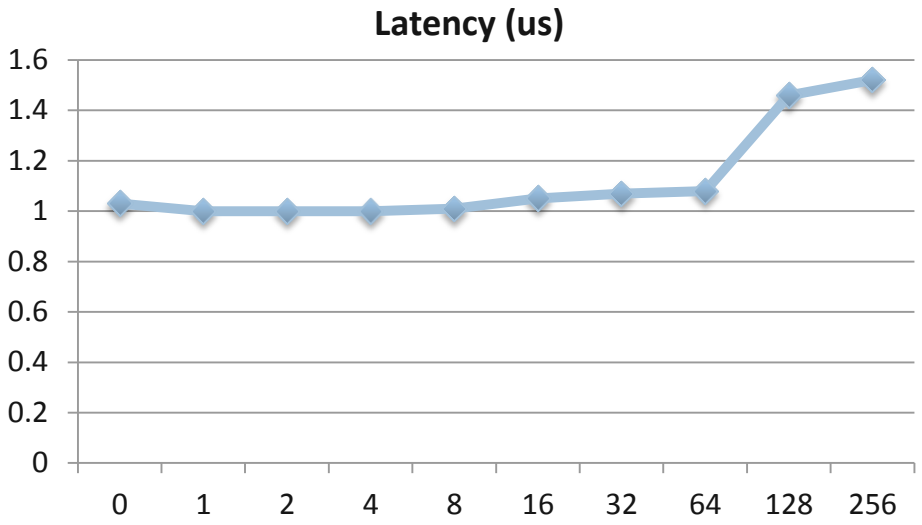
Does not cause significant overhead!

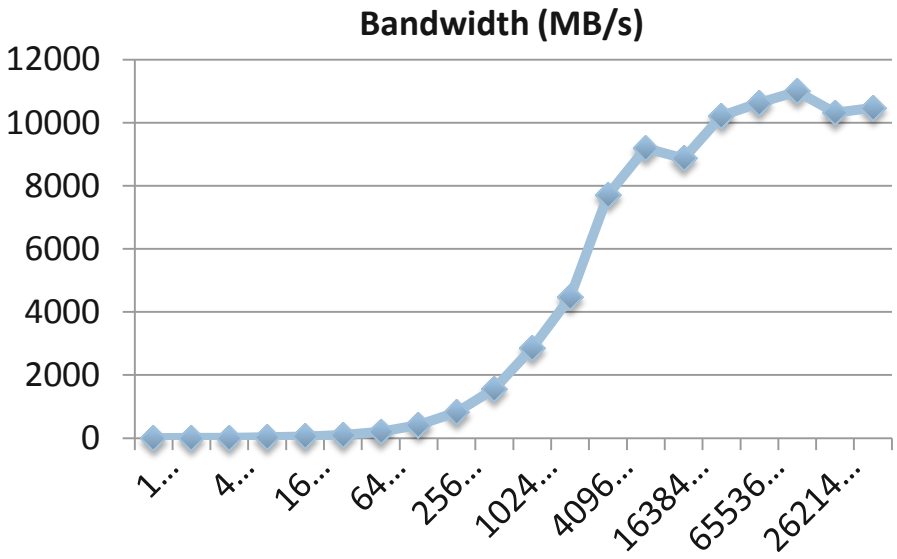


64 processes (ppn=8)

MXM Support in MPICH

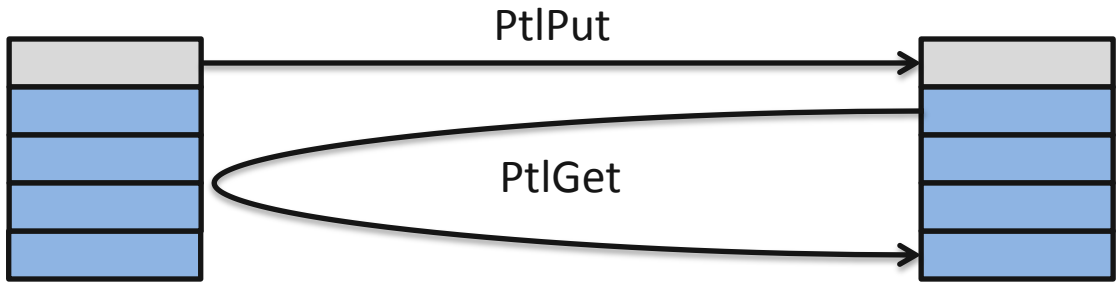
- MXM: Mellanox Messaging Accelerator
  - Supports multiple transports: RC, DC and UC
  - Intra-node: shared memory
  - Tag matching on receive side
- MXM netmod:
  - New since MPI 3.2
  - *Thanks to Mellanox for the contribution!*
- Test-machines
  - Mellanox Connect-IB EDR Infiniband
  - OFED-3.1-1.0.5





Portals 4 netmod in MPICH

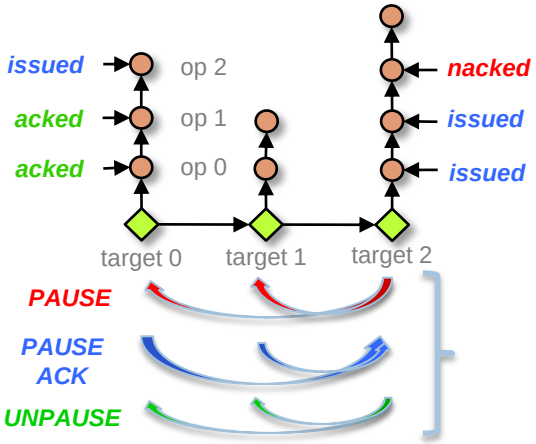
- Portals 4:
  - Connectionless, hardware independent network API
- Portals 4 netmod in MPICH 3.2:
  - Relies on Portals 4 MEs (matching list entries)
    - Matched message queues implemented in hardware
  - Messaging design
    - Messages <= eager threshold in single PtlPut operation
    - Hybrid Put/Get protocol for larger messages (receiver gets the rest of the data)
    - Support for messages greater than implementation defined max_msg_size



Portals 4 - Reliability

- Portals 4 only offers semi-reliability
  - Packets dropped on the network are retransmitted
  - Packets dropped on the end-host are not retransmitted
- Multiple scenarios can cause Portals 4 to go into flow control state
  - Event queue overflow
  - Unexpected buffer exhaustion
- MPICH adds a reliability layer (rportals)
  - Mandatory logging of all operations
  - Uses a separate EQ for origin side events
    - Queues operations if they will overflow the local EQ
    - Avoids silently dropping ACK packets
  - Recovery from flow control events
    - Global protocol to quiesce the network in rportals
      - Pause/Pause Ack/Unpause
    - NACKed operations are re-issued

Ken Raffenetti
Software Development Specialist



Explicit flow-control when hardware cannot handle messages

MPI-4 Fault Tolerance (User Level Failure Mitigation)

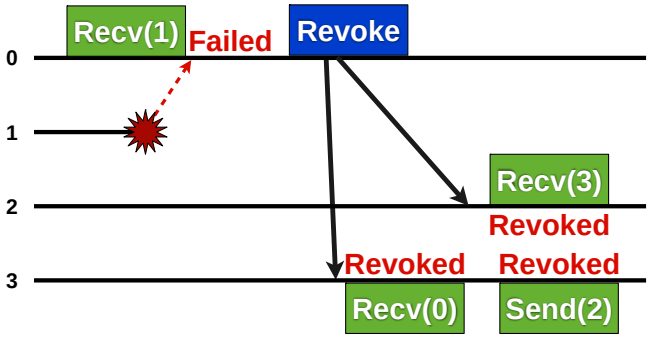
- Enable application-level recovery by providing minimal FT API to prevent deadlock and enable recovery
- Don't do recovery for the application, but let the application (or a library) do what is best.
- Only handling process failures currently

Wesley Bland
Software Developer (Intel)

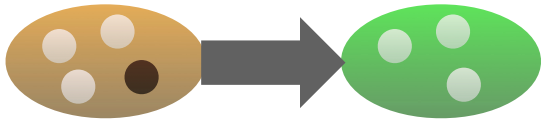
Failure Notification

MPI_ERR_PROC_FAILED

Failure Propagation



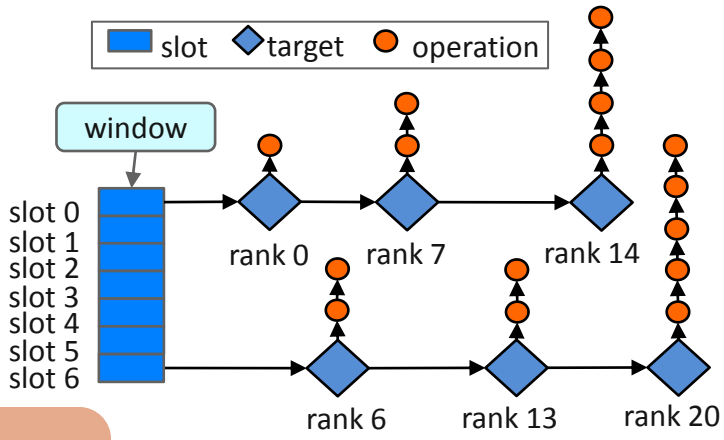
Failure Recovery



MPI_COMM_SHRINK()

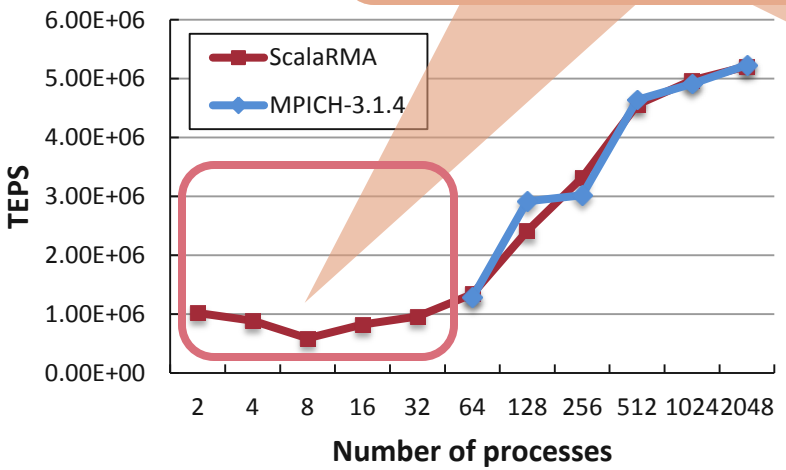
RMA: Sustainable Resource Management

- **Maintain operation metadata in a scalable data structure (RMA table)**
  - Total memory usage for operation and target metadata is constant
  - Strategies to restore resources

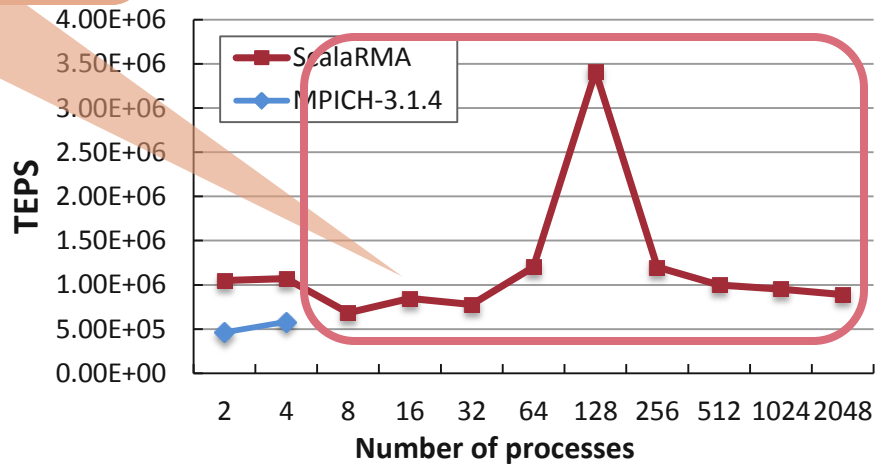


RMA table

mpich-3.1.4 runs out of memory when #op is large, whereas mpich-3.2 can complete



Strong scaling for graph500 benchmark (problem size: 2^{22} vertices), running with MXM

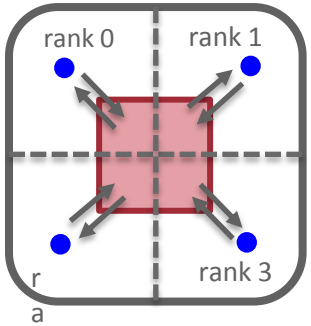


Weak scaling for graph500 benchmark (problem size: 2^{19} to 2^{29} vertices), running with MXM

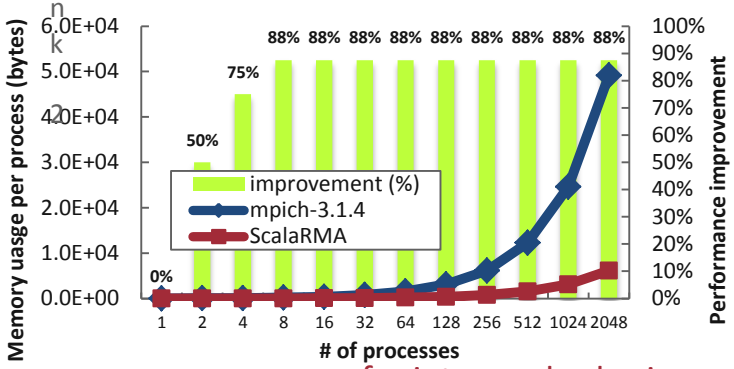
(Fusion cluster: 36GB memory per node with InfiniBand QDR interconnect)

Scalable Runtime Design

- **Window Metadata sharing and accessing**
  - **Window metadata**: base addresses, scaling unit sizes
  - Processes within one node share the same metadata copy
  - Memory usage is O(#nodes)
  - Communication cost of fetching metadata is constant (irrelevant with #P)
- **Scalable synchronization algorithm**
  - Active target: use Barrier-based algorithm for Fence when P is large, avoid O(P) memory
  - Passive target:
    - Avoid unlimited memory usage for lock queue on target
    - Avoid O(P) target metadata on origin
- **Streaming large ACC-like operations**
  - Avoid large memory consumption for temporary buffer
  - Overlap data transmission with computation

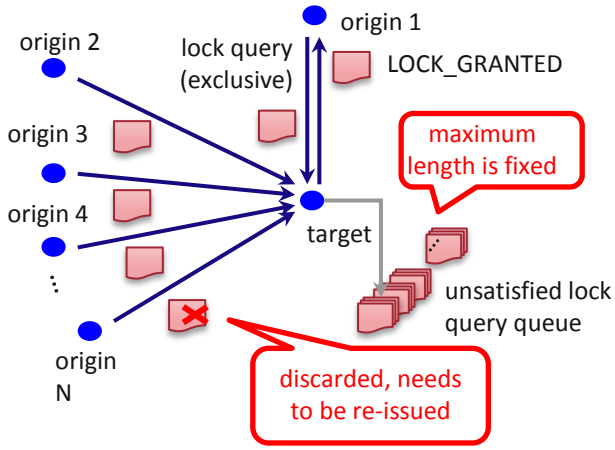


Intra-node: direct memory access

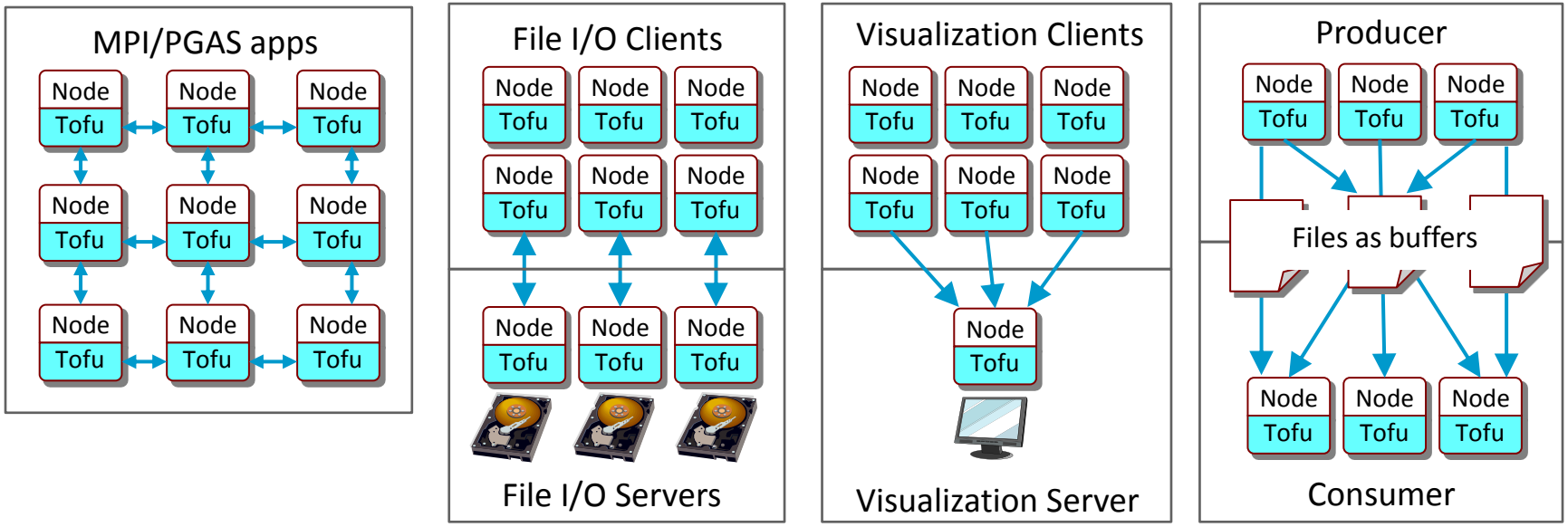


memory usage for intra-node sharing

(Fusion cluster: 36GB memory per node with InfiniBand QDR interconnect)



LLC: Network Abstractions for the Broader Computing (RIKEN and Fujitsu)



Architecture/programming model is changing

Network architecture

- Higher radix, more shared links

Node architecture

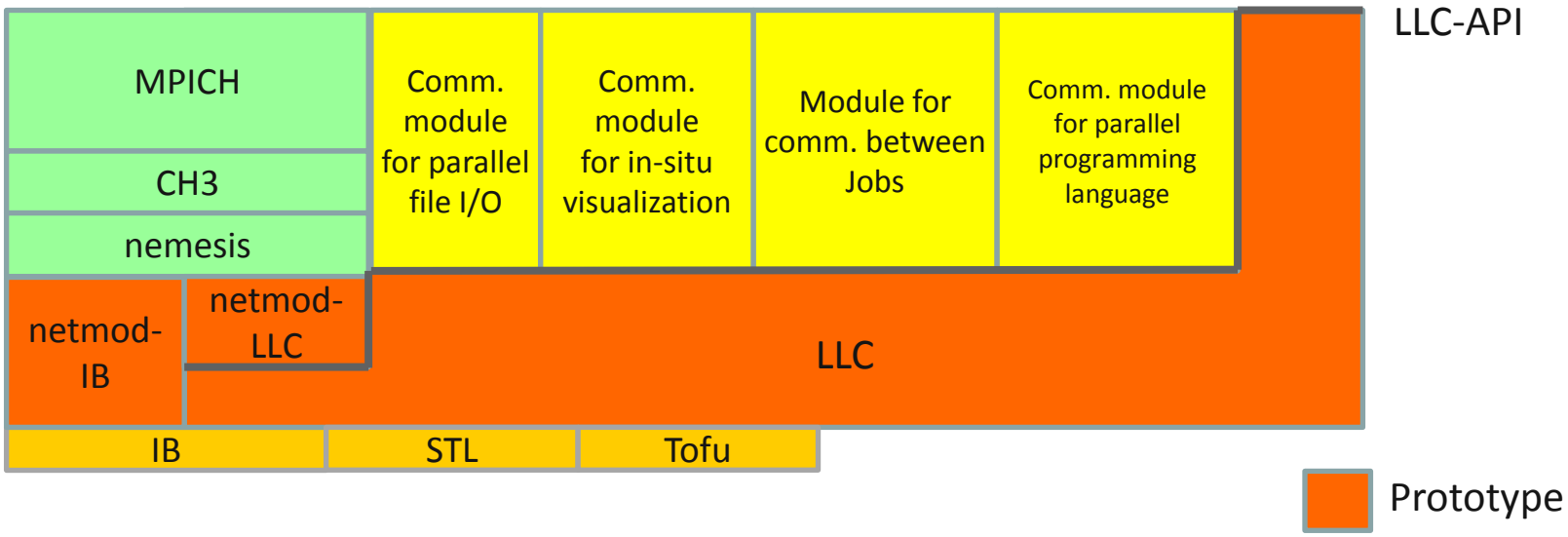
- More cores, less memory per core

Application

- Wide variety of communication models
  - Different programming model (e.g. MPI, PGAS), I/O clients, In-situ visualization, scientific work-flow based applications

Approach

Communication library for next generation computation and communication architecture



Key Capabilities

- Ability to deal with high parallelism on node (multicore/many-core architectures) and on the network (multiple DMA engines and shared communication paths)
- Explicit and introspective resource management (memory is the primary resource today, but network flow-control credits, ability for cache injection, etc., will be considered based on vendor roadmaps)

MPICH/LLC

- Initial integration of LLC support as a netmod inside MPICH/CH3
- Thanks to RIKEN for their contribution!
- Aimed for both the Post T2K machine and the Post K machine in Japan

MPICH/OFI

- Initial implementation within MPICH/CH3
- Aimed at understanding what OFI needs and what needs to change inside MPICH for this
- Fairly stable, but a few tests still fail

Argonne NATIONAL LABORATORY

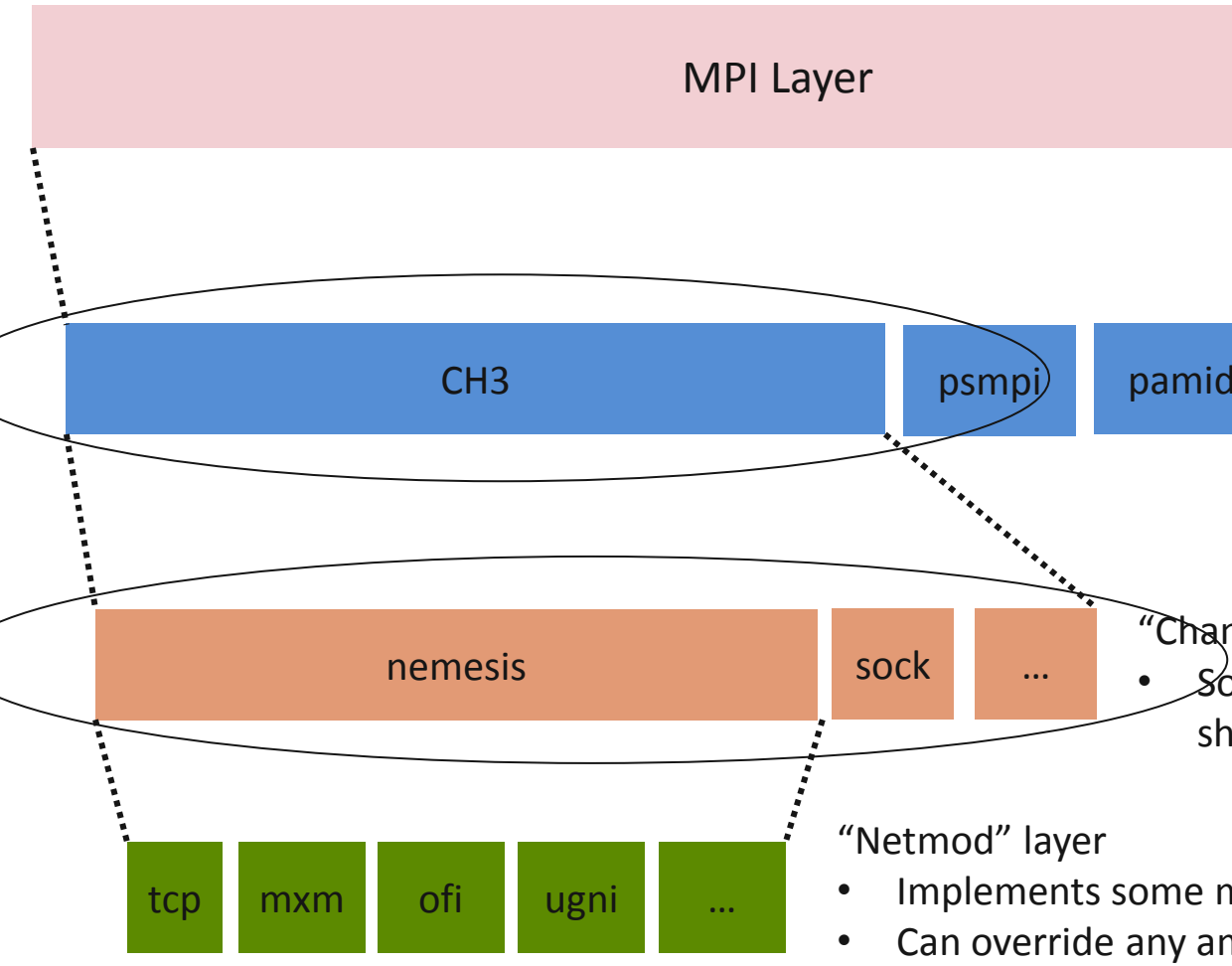
Planned Features for MPICH-3.3 (preview release at SC16)

U.S. DEPARTMENT OF ENERGY

MPICH-3.3 Feature Plan

- Announcing the CH4 Device!
  - Replacement for CH3, but we will maintain CH3 till all of our partners have moved to CH4
  - Two primary objectives:
    - Low-instruction count communication
      - Ability to support high-level network APIs
      - E.g., tag-matching in hardware, direct PUT/GET communication, ability to work with derived datatypes
    - Support for very high thread concurrency
      - Improvements to message rates in highly threaded environments (MPI_THREAD_MULTIPLE)
      - Move away from the LOCK/WORK/UNLOCK model to a much more scalable ENQUEUE/DEQUEUE model
      - Support for multiple network endpoints (THREAD_MULTIPLE or not)

MPICH layered structure: Current and Planned



Platform independent code
- Collectives
- Communicator management

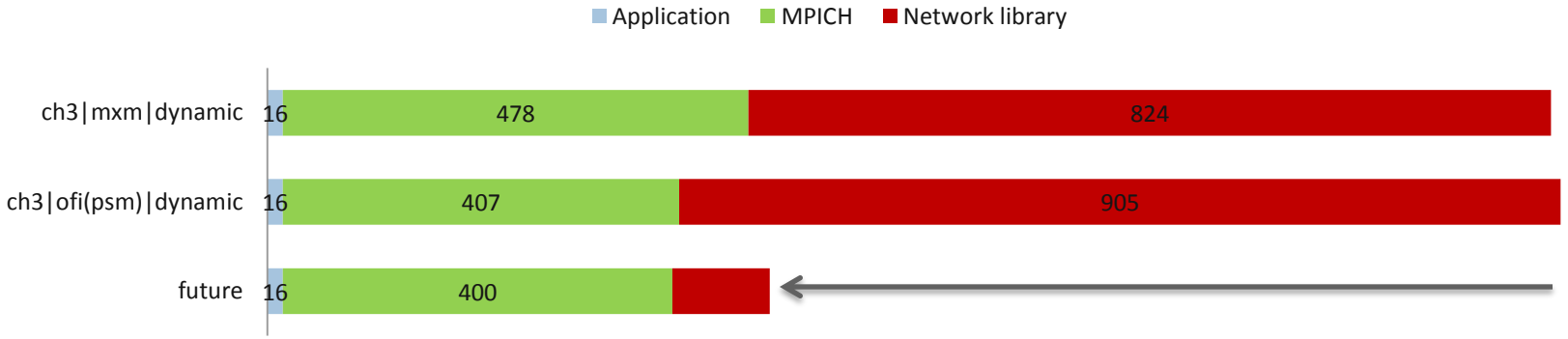
"Device" layer: Machine-specific functionality
- Derived datatype management
- RMA

"Channel" layer
- Some "core" functionality, such as shared memory

"Netmod" layer
- Implements some mandatory functionality
- Can override any amount of optional functionality (better bcast, better barrier)

Instruction count motivation

- With MPI features baked into next-generation hardware, we anticipate network library overheads will dramatically reduce.



- Message rate will come to be dominated by MPICH overheads

CH3 Shortcomings

Netmod API

- Passes down limited information and functionality to the network layer
  - `SendContig`
  - `SendNoncontig`
  - `iSendContig`
  - `iStartContigMsg`
  - `...`

Active Message Design

- All communication involves a packet header + message payload
  - Requires a non-contiguous memory access for all messages
- Workaround for Send/Recv override exists, but was somewhat clumsy add-in

Singular Shared Memory Support

- Performant shared memory communication centrally managed by Nemesis
- Network library shared memory implementations are not well supported
  - Inhibits collective offload

Function Pointers Not Optimized By Compiler

```
if (vc->comm_ops && vc->comm_ops->isend){
    mpi_errno =
        vc->comm_ops->isend(vc, buf, count, ...)
    goto fn_exit;
}
```

Non-scalable Virtual Connections

- 480 bytes * 1 million procs = 480MB(!) of VCs per process
- Connection-less networks emerging
  - VC and associated fields are overkill

CH4 Design Goals

High-Level Netmod API

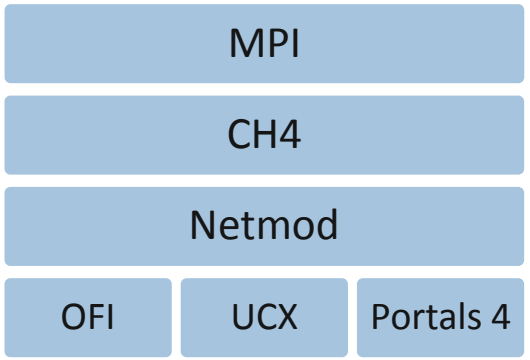
- Give more control to the network
  - `netmod_isend`
  - `netmod_irecv`
  - `netmod_put`
  - `netmod_get`
- Fallback to Active Message based communication when necessary
  - Operations not supported by the network

Provide default shared memory implementation in CH4

- Disable when desirable
  - Eliminate branch in the critical path
  - Enable better tuned shared memory implementations
  - Collective offload

"Netmod Direct"

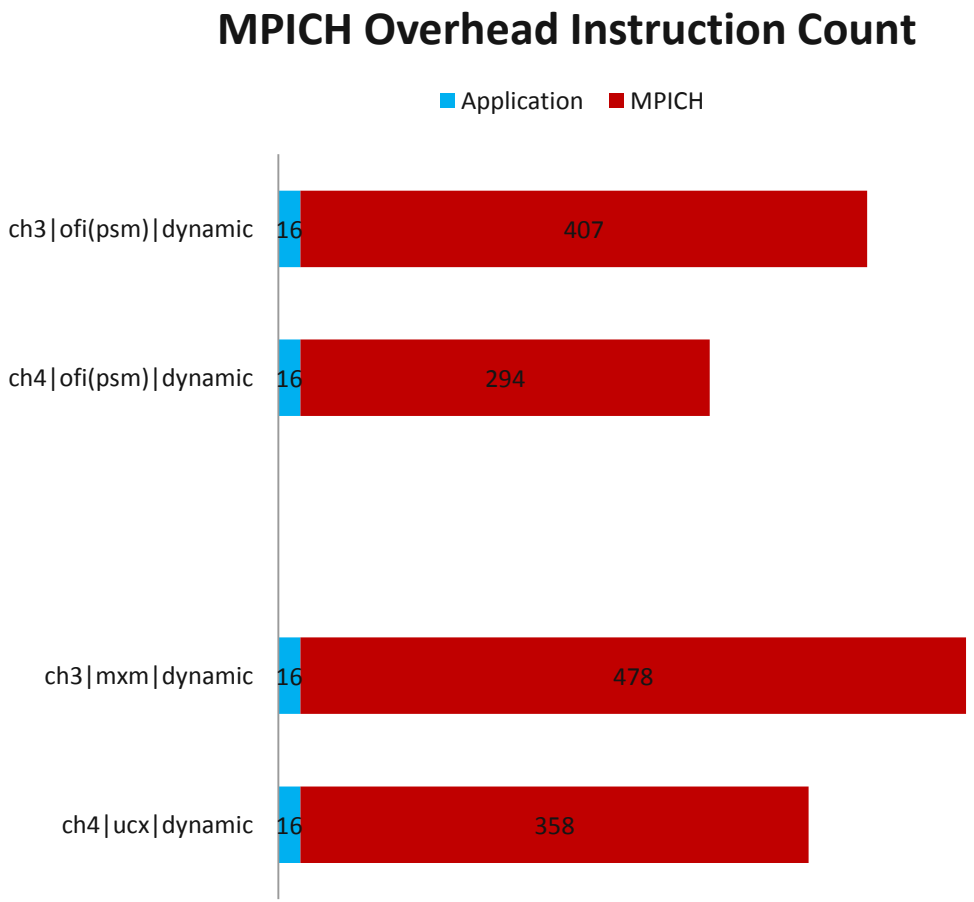
- Support two modes
  - Multiple netmods
    - Retains function pointer for flexibility
  - Single netmod with inlining into device layer
    - No function pointer



No Device Virtual Connections

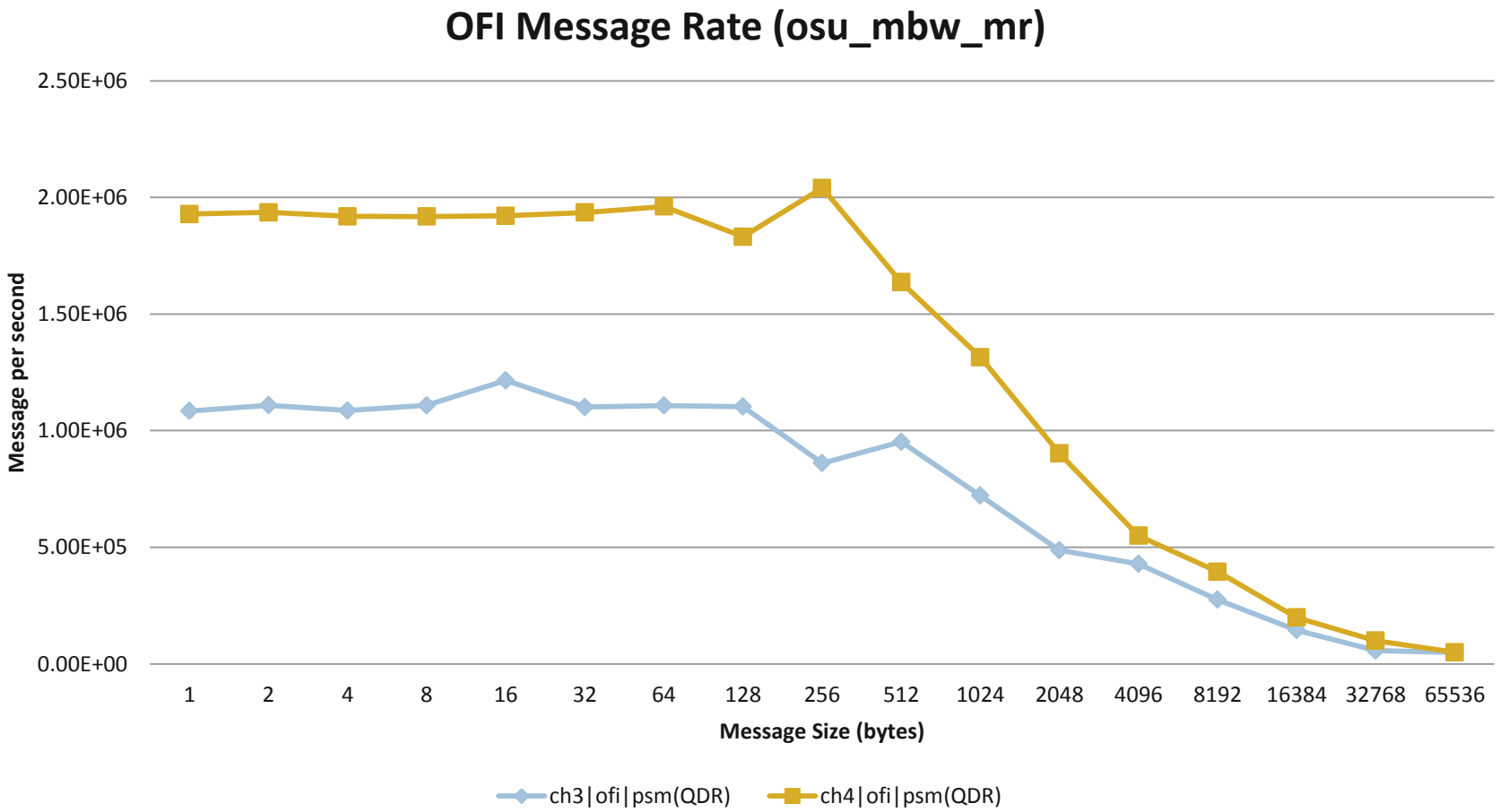
- Global address table
  - Contains all process addresses
  - Index into global table by translating `(rank+comm)`
- VCs can still be defined at the lower layers

Preliminary Improvements

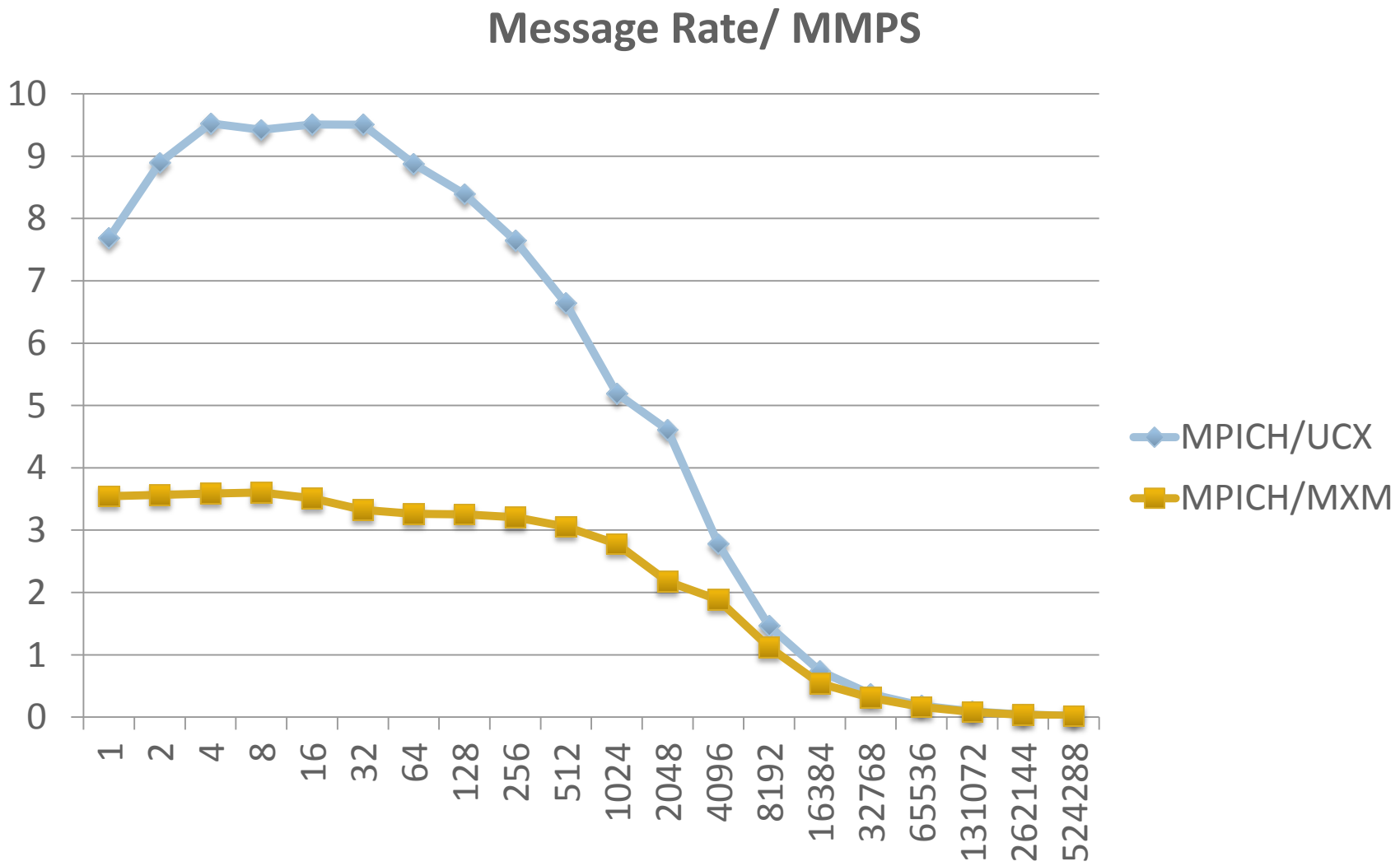


- 28% instruction count reduction with OFI
- 24% instruction count reduction with MXM/UCX

Preliminary Improvements with OFI

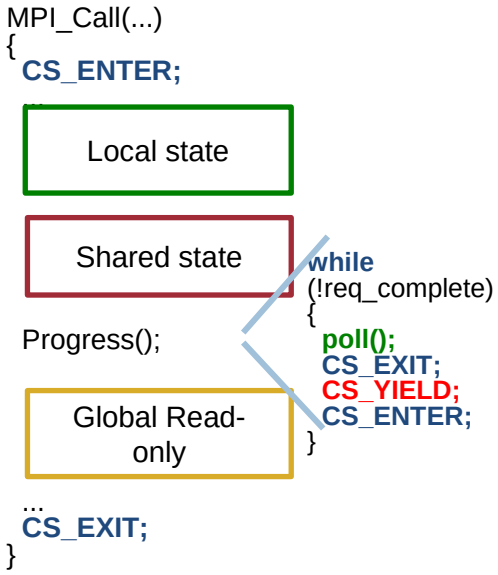


Preliminary Improvements



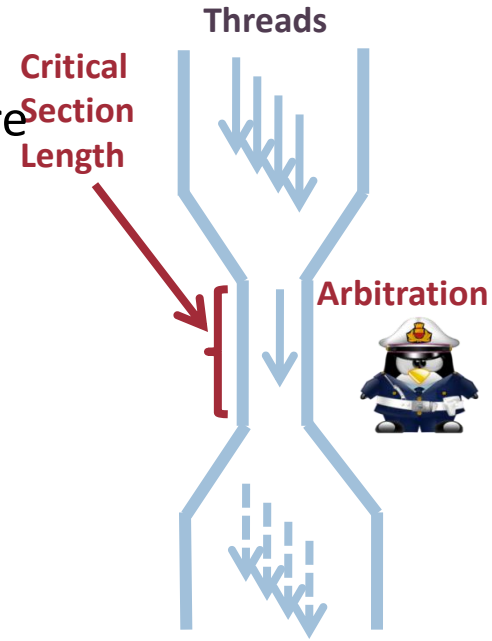
Multithreading Challenges and Aspects Being Considered

- **Granularity**
  - The current coarse-grained lock/work/unlock model is not scalable
  - Fully lock-free is not practical
    - Rank-wise progress constrains
    - Ordering constrains
    - Overuse of atomics and memory barriers can backfire



Abdelhalim Amer (Halim)

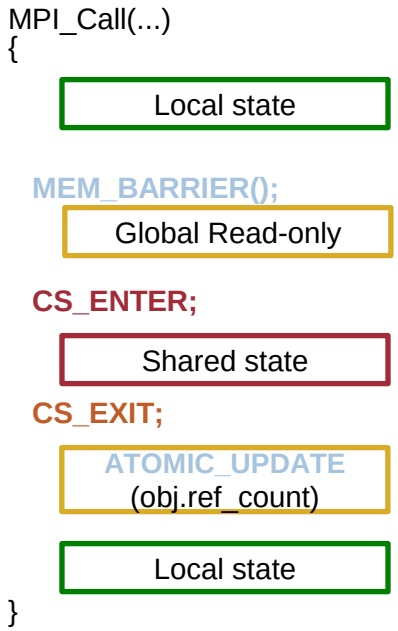
- **Arbitration**
  - Traditional Pthread mutex locking is biased by the hardware (NUCA) and the OS (slow wakeups)
  - Causes waste because of the lack of correlation between resource acquisition and work
- **Hand-off latency**
  - Slow hand-offs waste the advantages of fine-granularity and smart arbitration
  - Trade-offs must be carefully addressed



Granularity Optimization Being Considered

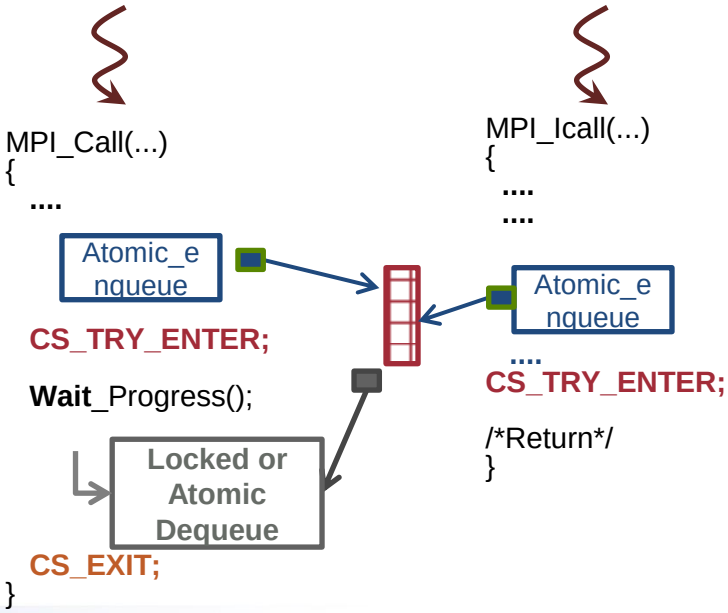
Combination of several methods to achieve fine granularity

- Locks and atomics are not always necessary
  - Memory barriers can be enough
  - E.g. only read barrier for MPI_Comm_rank
- Brief-global locking: only protect shared objects
- Lock-free wherever possible
  - Atomic reference count updates
  - Lock-free queue operations



Moving towards a lightweight queue/dequeue model

- Reduce locking requirements and move to a mostly lock-free queue/dequeue model
- Reduce unnecessary progress polling for nonblocking operations
- Enqueue operations all atomic
- Dequeue operations
  - Atomic for unordered queues (RMA)
  - Fine-grained locking for ordered queues
- The result is a mostly lock-free model for nonblocking operations

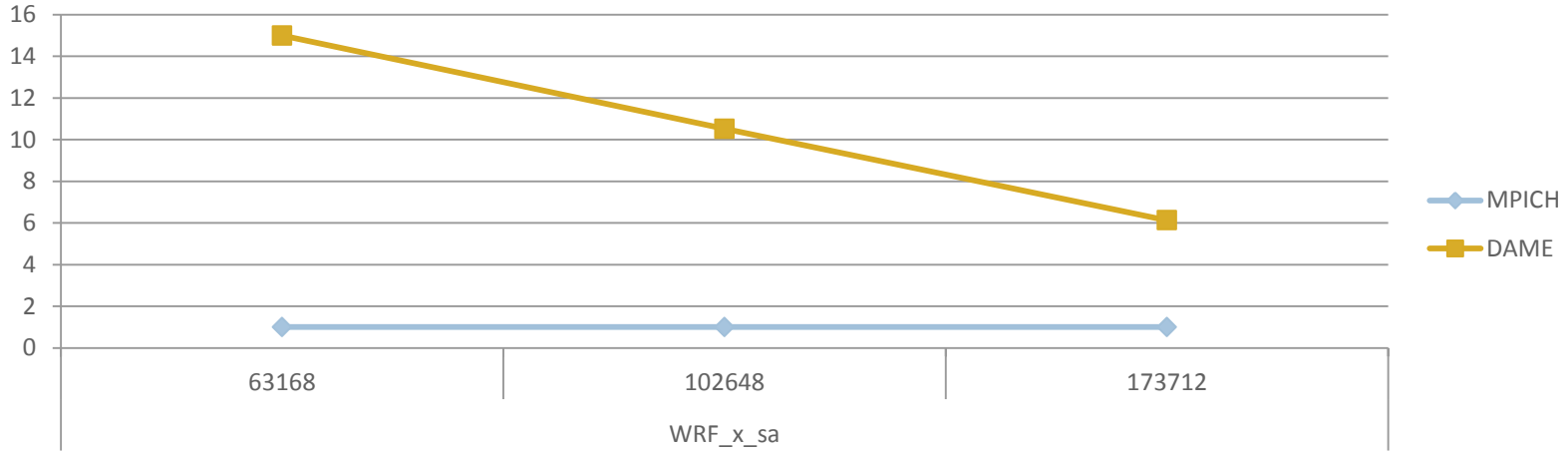


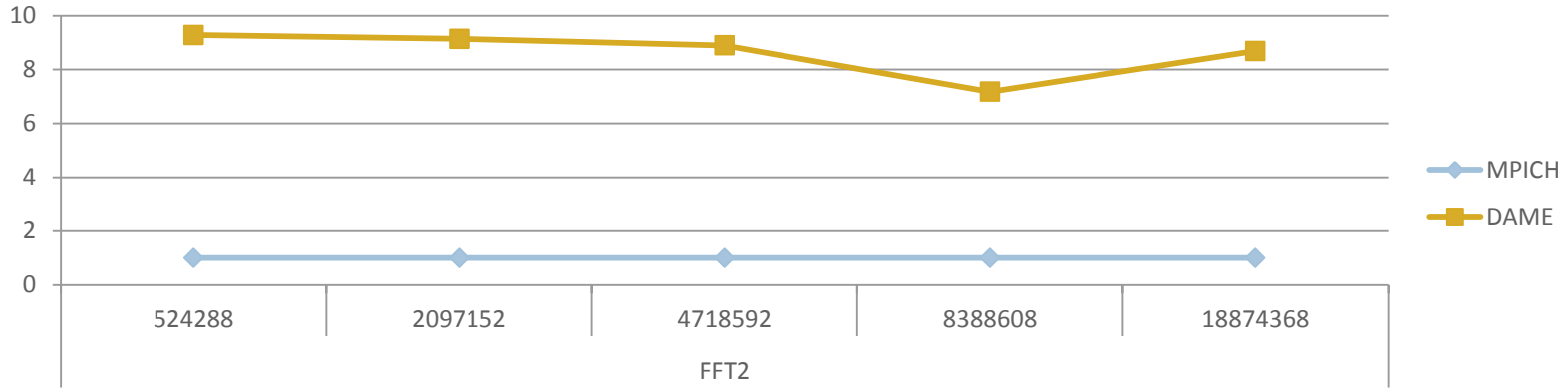
MPICH/CH4 Netmod plans

- Four initial open-source netmods planned
  - OFI (jointly with Intel)
  - UCX (jointly with Mellanox)
  - LLC (jointly with RIKEN)
  - Portals4
  - No plan to directly support TCP, since other netmods would support it
    - This might change if the support-level is not sufficient for us
  - Hackathons and code camps to help move these along
- Partner netmods
  - Cray uGNI/DMAPP netmod (closed source)
  - NUDT TH-2 netmod
- Non-netmod implementations
  - IBM, MVAPICH, ParTec, FG-MPI, …

DAME: a new engine for derived datatypes

- **Who:** Tarun Prabu, Bill Gropp (UIUC)
- **Why:** DAME is an improved engine for derived-datatypes
  - The Dataloop code (type processing today) effective, but requires many function calls (the “piece functions”) for each “leaf type”
  - Piece Functions (function pointers) are difficult for most (all?) compilers to inline, even with things like link-time optimizations
- **What:** DAME implements a new description of the MPI datatype, then transforms that description into efficient memory operations
- **Design Principles**:
  - Low processing overhead
  - Maximize ability of compiler to optimize code
  - Simplify partial packing
  - Enable memory access optimizations
- **Optimizations**:
  - Memory access optimizations can be done by shuffling primitives as desired. This is done at “commit” time.
  - Other optimizations such as normalization (e.g. an indexed with identical stride between elements), displacement sorting and merging can also easily be performed at commit-time.





Relative Communication Speedup over MPICH (p=2)

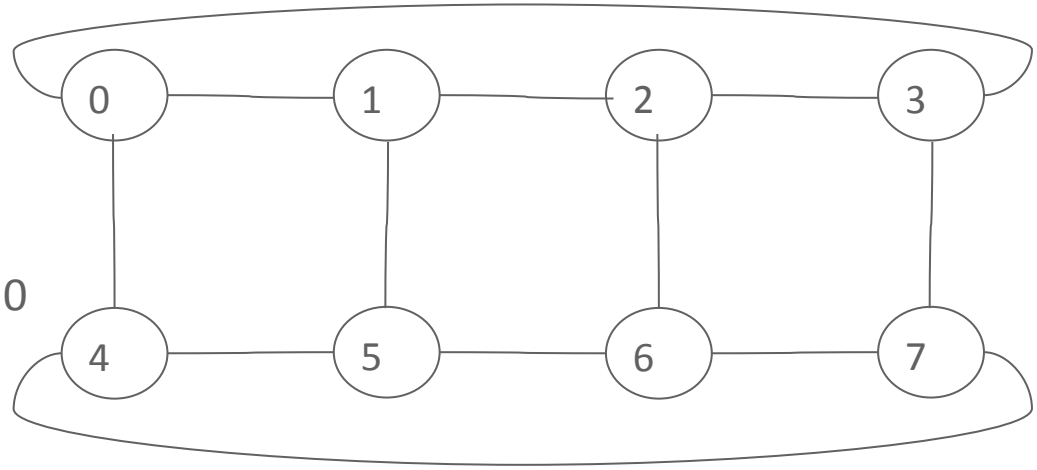
Tests from DDTBench

MPI Graph and Cartesian Topology Functions (II)

- **MPI_Cart_create(***comm_old*, *ndims*, *dims*, *periods*, *reorder*, *comm_cart* **)**
  - *comm_old* [in] input communicator without topology (handle)
  - *ndims* [in] number of dimensions of Cartesian grid (integer)
  - *dims* [in] integer array of size ndims specifying the number of processes in each dimension
  - periods [in] logical array of size ndims specifying whether the grid is periodic (true) or not (false) in each dimension
  - reorder [in] ranking may be reordered (true) or not (false) (logical)
  - comm_graph [out] communicator with Cartesian topology (handle)

Dimension	#Processes
1	4
2	2

ndims = 2
dims = 4, 2
periods = 1, 0

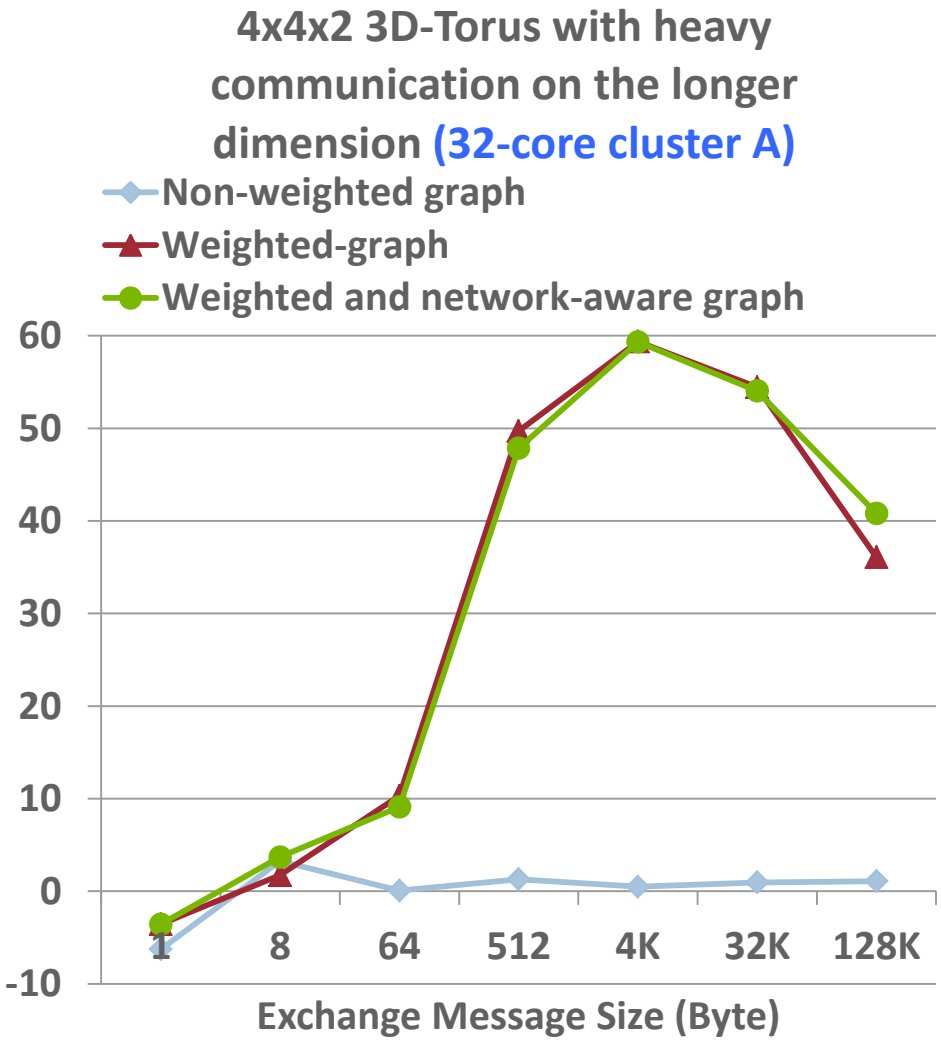


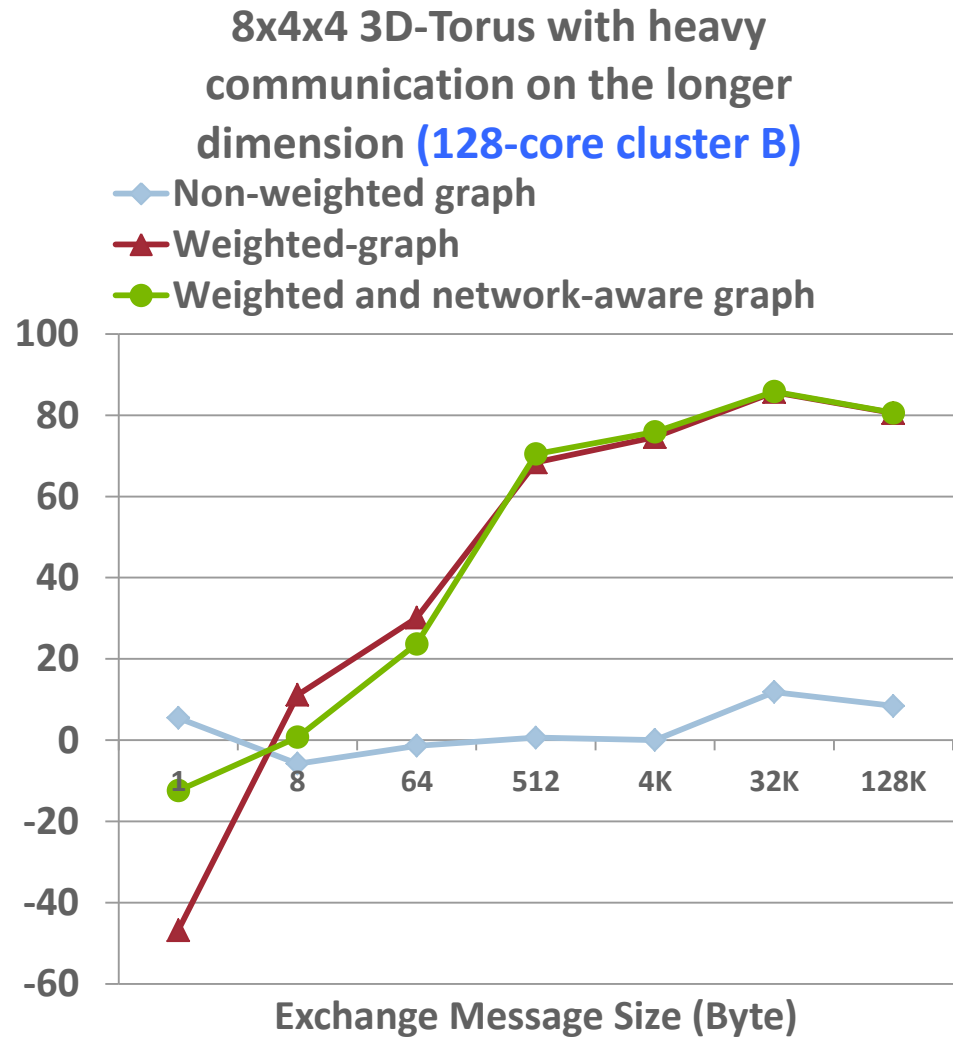
4x2 2D-Torus

Tools for Implementation of Topology Functions

- HWLOC library for extracting node architecture:
  - A tree architecture, with nodes at top level and cores at the leaves
  - Cores with lower-level parents (such as caches) are considered to have higher communication performance
- IB subnet manager (*ibtracert*) for extracting network distances:
  - Do the discovery offline, before the application run
  - Make a pre-discovered network distance file
- Scotch library for mapping virtual to physical topologies:
  - Source and target graphs are weighted and undirected
  - Uses recursive bi-partitioning for graph mapping

Exchange Micro-benchmark: Topology-aware Mapping Improvement over Block Mapping (%)



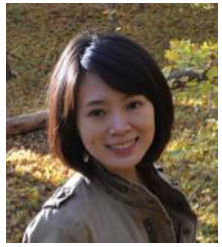


Argonne NATIONAL LABORATORY

Casper: Asynchronous Progress for MPI RMA

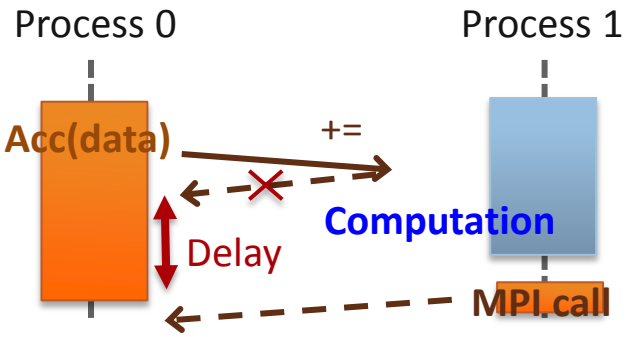
U.S. DEPARTMENT OF ENERGY

Asynchronous Progress in MPI RMA

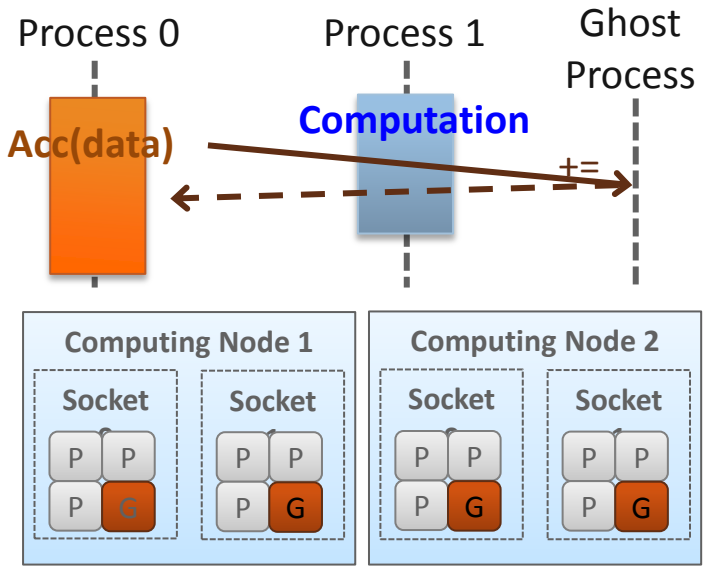


Min Si

- MPI RMA communication
  - Not truly one-sided
  - e.g., 3D accumulates of double precision data on RDMA* network
  - Traditional approaches
    - Thread-based
    - Interrupt-based



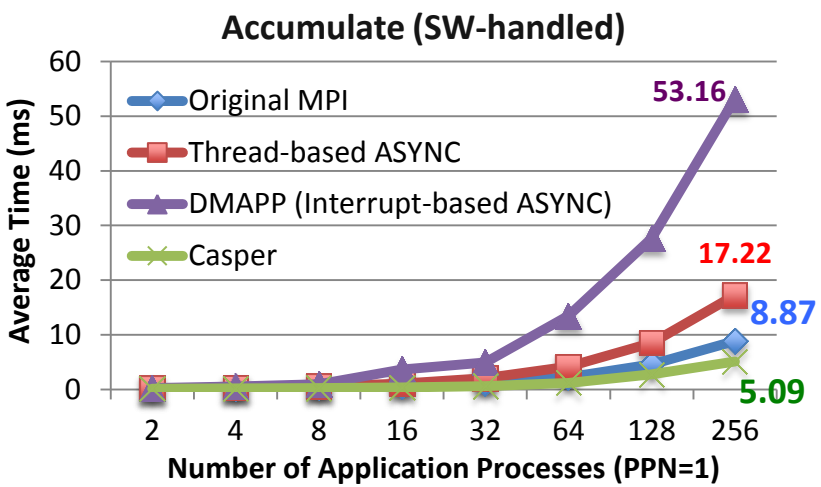
- Casper
  - Process-based Asynchronous Progress
  - Arbitrary #ghost processes [**Flexible**]
  - PMPI redirection [**Portable**]
  - Ghost process handles operations via MPI-3 shm-window.



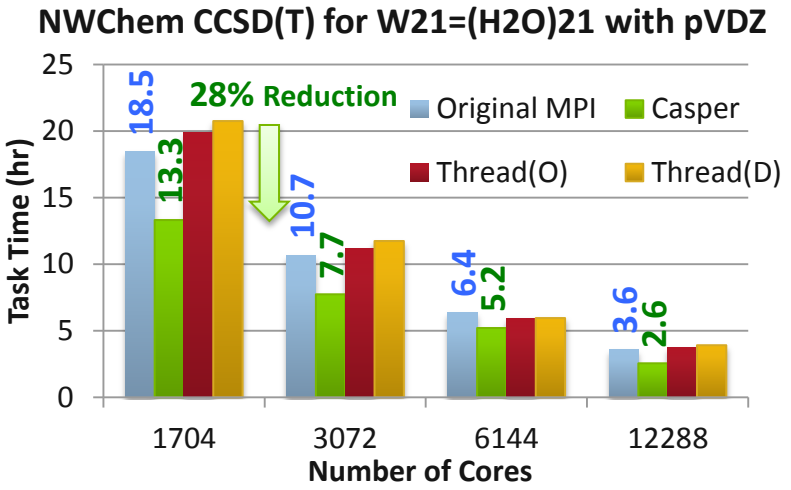
* **RDMA** : Remote Direct Memory Access

Casper Evaluation on Edison

- Efficient asynchronous progress for SW-handled operations



- NWChem
  - Quantum chemistry application suite



Software Release

[**Website**] www.mcs.anl.gov/project/casper
[**Download**] git clone http://git.mpich.org/soft/dev/casper.git
[**Support**] casper-users@lists.mpich.org

[**Beta Release**] January 2016 [**GA Release**] 2016-03-10
[**Point-of-Contact**] Min Si (msi@anl.gov) [**Partners**]

Programming Models and Runtime Systems Group

Group Lead

- Pavan Balaji (computer scientist and group lead)

Current Staff Members

- Abdelhalim Amer (postdoc)
- Yanfei Guo (postdoc)
- Rob Latham (developer)
- Lena Oden (postdoc)
- Ken Raffenetti (developer)
- Sangmin Seo (postdoc)
- ***Min Si (postdoc)***
- ***Min Tian (visiting scholar)***

Past Staff Members

- Antonio Pena (postdoc)
- Wesley Bland (postdoc)
- Darius T. Buntinas (developer)
- James S. Dinan (postdoc)
- David J. Goodell (developer)
- Huiwei Lu (postdoc)
- Yanjie Wei (visiting scholar)
- Yuqing Xiong (visiting scholar)
- Jian Yu (visiting scholar)
- Junchao Zhang (postdoc)
- Xiaomin Zhu (visiting scholar)

Current and Recent Students

- Ashwin Aji (Ph.D.)
- Abdelhalim Amer (Ph.D.)
- Md. Humayun Arafat (Ph.D.)
- Alex Brooks (Ph.D.)
- Adrian Castello (Ph.D.)
- Dazhao Cheng (Ph.D.)
- James S. Dinan (Ph.D.)
- Piotr Fidkowski (Ph.D.)
- Priyanka Ghosh (Ph.D.)
- Sayan Ghosh (Ph.D.)
- Ralf Gunter (B.S.)
- Jichi Guo (Ph.D.)
- Yanfei Guo (Ph.D.)
- Marius Horga (M.S.)
- John Jenkins (Ph.D.)
- Feng Ji (Ph.D.)
- Ping Lai (Ph.D.)
- Palden Lama (Ph.D.)
- Yan Li (Ph.D.)
- Huiwei Lu (Ph.D.)
- Jintao Meng (Ph.D.)
- Ganesh Narayanaswamy (M.S.)
- Qingpeng Niu (Ph.D.)
- Ziaul Haque Olive (Ph.D.)
- David Ozog (Ph.D.)
- Renbo Pang (Ph.D.)
- Sreeram Potluri (Ph.D.)
- Li Rao (M.S.)
- Gopal Santhanaraman (Ph.D.)
- Thomas Scogland (Ph.D.)
- Min Si (Ph.D.)
- Brian Skjerven (Ph.D.)
- Rajesh Sudarsan (Ph.D.)
- Lukasz Wesolowski (Ph.D.)
- Shucai Xiao (Ph.D.)
- Chaoran Yang (Ph.D.)
- Boyu Zhang (Ph.D.)
- Xiuxia Zhang (Ph.D.)
- Xin Zhao (Ph.D.)

Advisory Board

- Pete Beckman (senior scientist)
- Rusty Lusk (retired, STA)
- Marc Snir (division director)
- Rajeev Thakur (deputy director)

Argonne NATIONAL LABORATORY

: http://www.mcs.anl.gov/~balaji Email: balaji@anl.gov

Group website: http://www.mcs.anl.gov/group/pmrs/

U.S. DEPARTMENT OF ENERGY

Cray MPI Update

SC15 – MPICH BOF

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, URIKA and YARCDATA. The following are trademarks of Cray Inc.: ACE, APPRENTICE2, CHAPEL, CLUSTER CONNECT, CRAYPAT, CRAYPORT, ECOPHLEX, LIBSCI, NODEKARE, THREADSTORM. The following system family marks, and 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 names and brands may be claimed as the property of others. Other product and service names mentioned herein are the trademarks of their respective owners.

Cray MPI Highlights Since SC14

- Current release: MPT 7.2.6 (November 2015)
- Merge to ANL MPICH 3.1.2 release
- MPI-3 Fortran 2008 language bindings are supported for the Cray CCE Fortran compiler
- MPI-3 Tools Interface for Cray environment variables
- GPU-to-GPU support for MPI-3 RMA

Cray MPI Highlights Since SC14 (continued)

- MPI-IO collective buffering support for Lustre file locking mode
- MPI collective improvements
- MPICH ABI compatibility support for Intel MPI 5.x (mainly changes to cray-mpich-abi module)
- MPI_Alloc_mem can now return memory that is backed by hugepages

Future Plans

- MPI 3.1 support (Merge with MPICH 3.2)
- Additional MPI_THREAD_MULTIPLE optimizations
- Optimizations to support future processors on Cray XC systems
- Investigate/move to MPICH CH4

MPI @ Intel

William R. Magro
Intel Fellow & Chief Technologist, HPC Software
Intel Corporation
SC'15 MPICH BoF
November 17, 2015

Intel® MPI Library & MPICH

Intel MPI 5.1.2 is based on MPICH-3.1.2

- Intel MPI 5.2 will migrate to MPICH-3.2 (Q2 2016)

Intel is part of the MPICH ABI Compatibility Initiative (member since inception)

Contributions to Argonne MPICH this year:

- Netmod support for OFI in CH3
- Hydra hierarchy
- Wait mode support

Intel® MPI Library: New Features

Intel MPI Version 5.1.2 released with new features:

- Demonstrated scaling to 340,000 ranks with HPLinpack
- Latest Intel® Xeon Phi™ processor support (code-named Knights Landing)
- Intel® Omni-Path Architecture support
- OpenFabrics Interfaces (OFI / libfabric) support
- YARN* process manager support (Hadoop compatibility)
- Lustre* support on Intel® Xeon Phi™ coprocessor
- GPFS* support
- ILP64 modules in Fortran 90

Next Generation MPICH: CH4

Open-source implementation based on MPICH

- Uses the new CH4 infrastructure
  - Contributing to design of CH4 API to minimize software overhead
- Targets existing and new fabrics via next-gen Open Fabrics Interface (OFI)
  - Ethernet/sockets, InfiniBand*, Cray Aries*, IBM BG/Q, Intel® Omni-Path

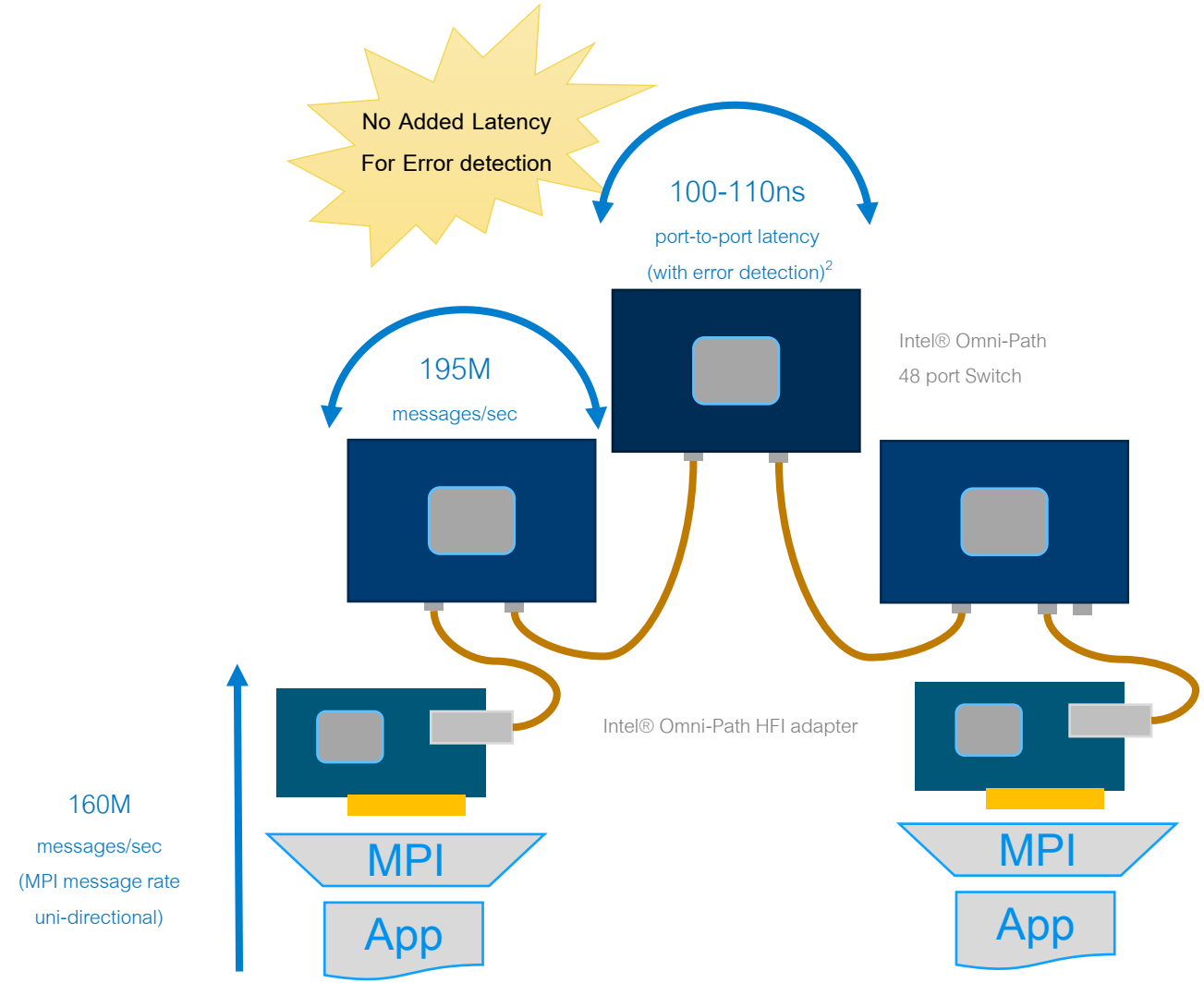
Early proof points:

- MPICH/CH4/OFI over sockets provider tested to 4K ranks
- Scaling to >1 million ranks on IBM BG/Q

Intel® MPI Library in the Cloud/Big Data

- Intel MPI Library supported in Microsoft* Azure Cloud Linux A8 and A9 instances
- Integrating Hadoop* and MPI: data management capabilities of Hadoop with the performance of native MPI applications on the same cluster
  - YARN/Intel MPI prototype to support Hadoop integration for Cloudera*
- Compatible with Intel® Data Analytics Acceleration Library (Intel® DAAL)

Intel® Omni-Path Architecture: Accelerating Data Movement Through the Fabric

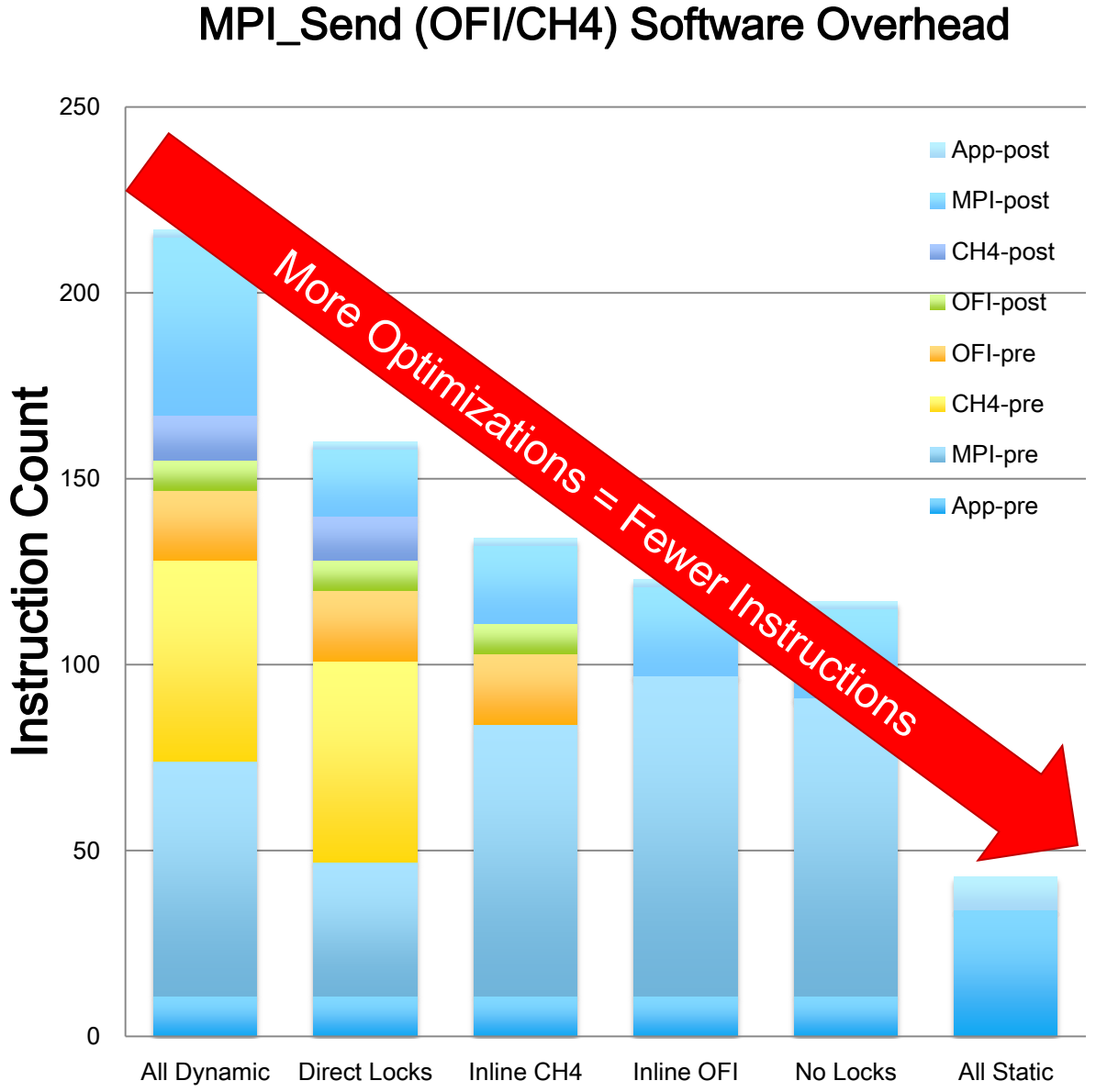


[1] Based on Intel projections for Wolf River and Prairie River maximum messaging rates, compared to Mellanox CS7500 Director Switch and Mellanox ConnectX-4 adapter and Mellanox SB7700/SB7790 Edge switch product briefs posted on www.mellanox.com as of November 3, 2015.
[2] Latency reductions based on Mellanox CS7500 Director Switch and Mellanox SB7700/SB7790 Edge switch product briefs posted on www.mellanox.com as of November 3, 2015, compared to Intel measured data that was calculated from difference between back to back osu_latency test and osu_latency test through one switch hop. 10ns variation due to "near" and "far" ports on an Intel® OPA edge switch. All tests performed using Intel® Xeon® E5-2697v3 with Turbo Mode enabled.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. Copyright © 2015, Intel Corporation.

CH4 Over OFI

- Maturing: ~92% test pass rate
- Supports OFI networks:
  - InfiniBand Architecture*
  - Intel* Omni-Path Architecture
  - IBM BG/Q*
  - Cray Aries*
  - Ethernet/sockets
- Very low instruction counts
  - **43 instructions** from application to OFI with all optimizations



Data source: Intel

Legal Disclaimer & Optimization Notice

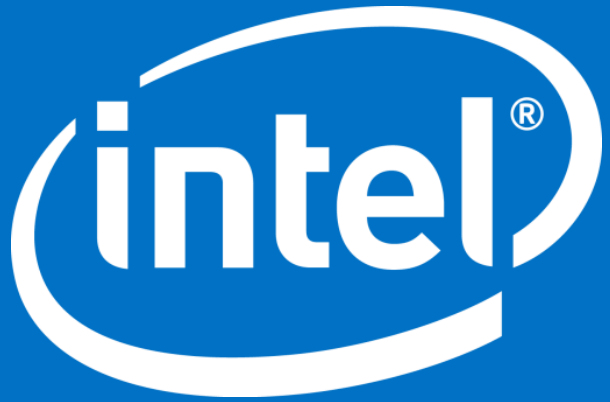
INFORMATION IN THIS DOCUMENT IS PROVIDED “AS IS”. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

Optimization Notice

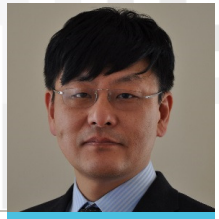
Intel’s compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804


intel

Lenovo MPI view

Chulho Kim, HPC Software Development Architect



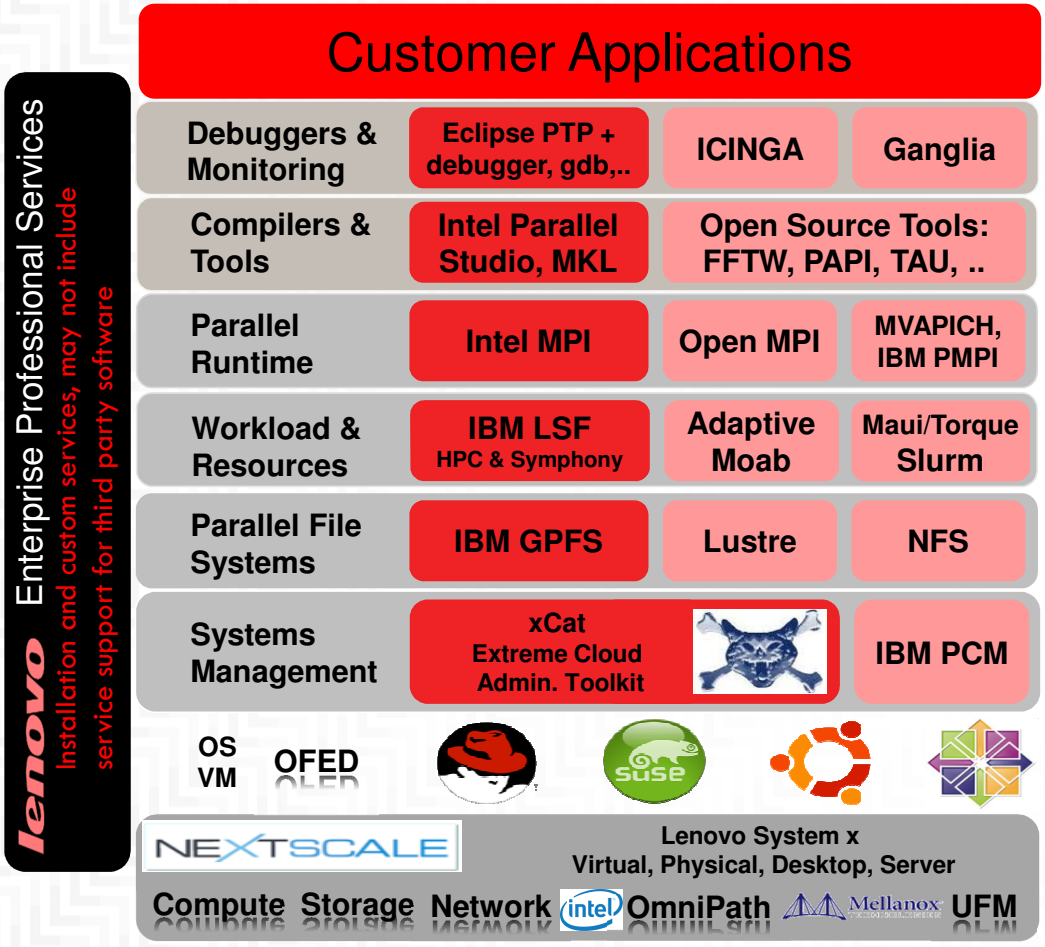
@chk12603

Super Computing 2015 – MPICH BOF – November 17th 2015

Lenovo's HPC Industry Standards Participation

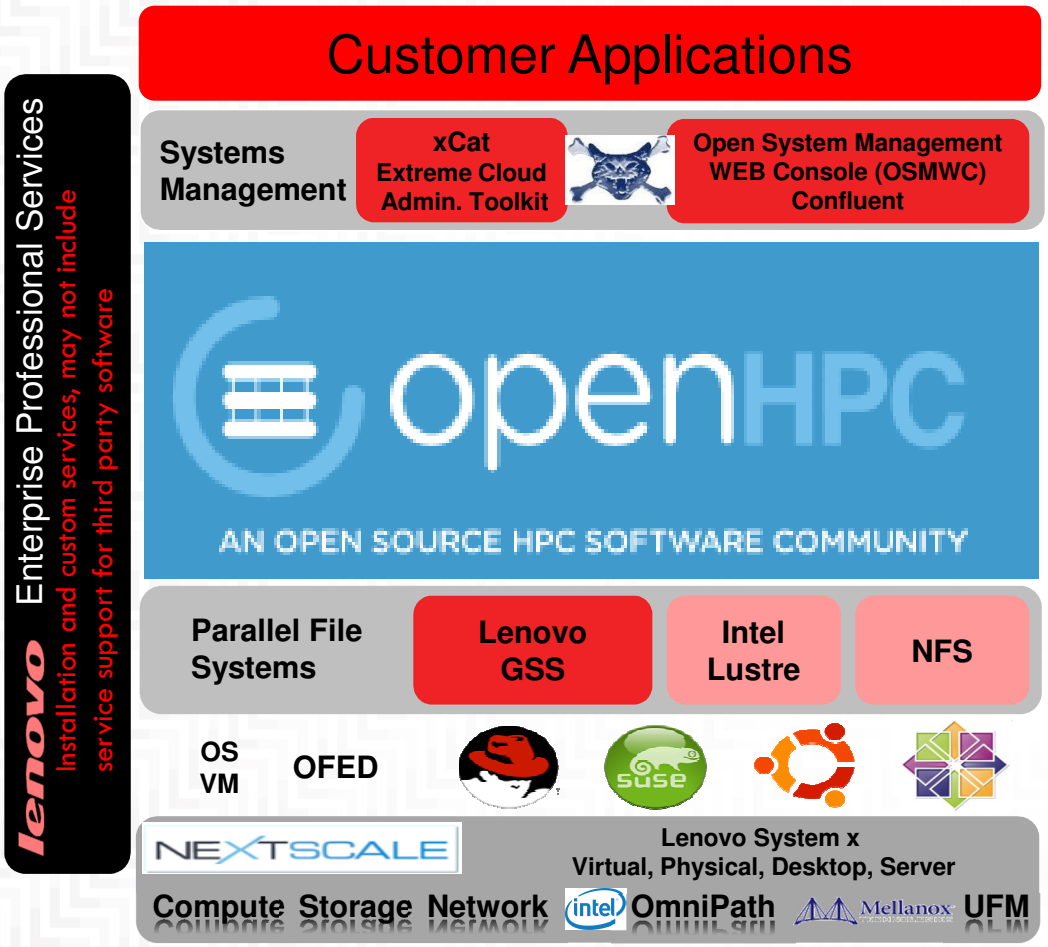
- **MPI 3.1/4.0 Standards**
  - Fully participating member since December of 2014 (1st meeting after becoming Lenovo)
  - 4 meetings a year. Helped approve MPI 3.1 standard and now starting to look at MPI 4.0 efforts.
- **InfiniBand Trade Association (IBTA)**
  - Lenovo is member of IBTA
- **OpenFabric Alliance (OFA)**
  - Lenovo is member of OFA
- **Linux Foundation and HPC Communities openHPC initiative**
  - **Founding member – will actively participate in helping define/validate the software stack, especially on system management and MPI runtime**

HPC Software Solutions through Partnerships



- **Building Partnerships to provide the "Best In-Class" HPC Cluster Solutions for our customers**
- Collaborating with software vendors to provide features that optimizes customer workloads
- Leveraging "Open Source" components that are production ready
- Contributing to "Open Source" (i.e. xCAT, Confluent, OpenStack) to enhance our platforms
- Providing "Services" to help customers deploy and optimize their clusters

Future High Performance Computing Open Solutions



- **Partnering as founding member of OpenHPC initiative to establish a common Open HPC Framework**
- **Collaborating with Oxford University to create an Open System Management framework for small to medium clusters**
- **In openHPC environment, Lenovo will focus on MPICH CH4 enablement on both Intel and Mellanox interconnects with Application binary runtime compatibility with Intel MPI Runtime.**

THANK YOU

DAKUJEM DANK BEDANKT MERCI TAKK 谢谢
ありがとう СПАСИБО GRACIAS DZIĘKUJĘ DANKE
OBRIGADO БЛАГОДАРЯ GRAZIE הדות GRACIAS

Lenovo™


Lenovo™

Mellanox MPICH Status

Super Computing 2015

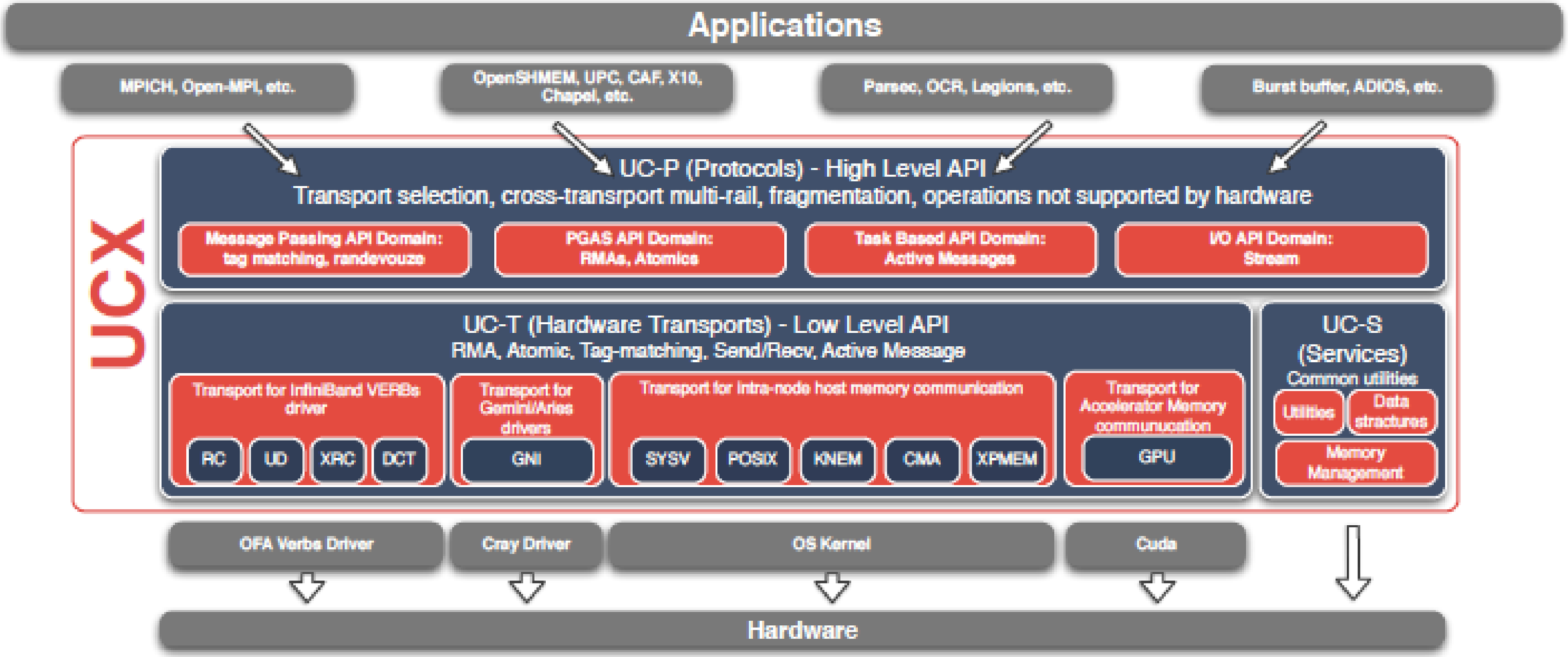
Mellanox TECHNOLOGIES

Connect. Accelerate. Outperform.™

UCX Overview

- UCX is the next generation open-source production-grade communication middleware.
- Collaboration between industry, laboratories, and academia.
- Exposes near bare-metal performance by reducing software overheads.
- Provides both a thin, low-level API, and a general purpose high-level API.
- Targeted at message passing (MPI) and PGAS (e.g. OSHMEM, UPC) programming models, as well as emerging programming models.
- UCX support has been integrated into MPICH, Open MPI, and Open MPI based OSHMEM.

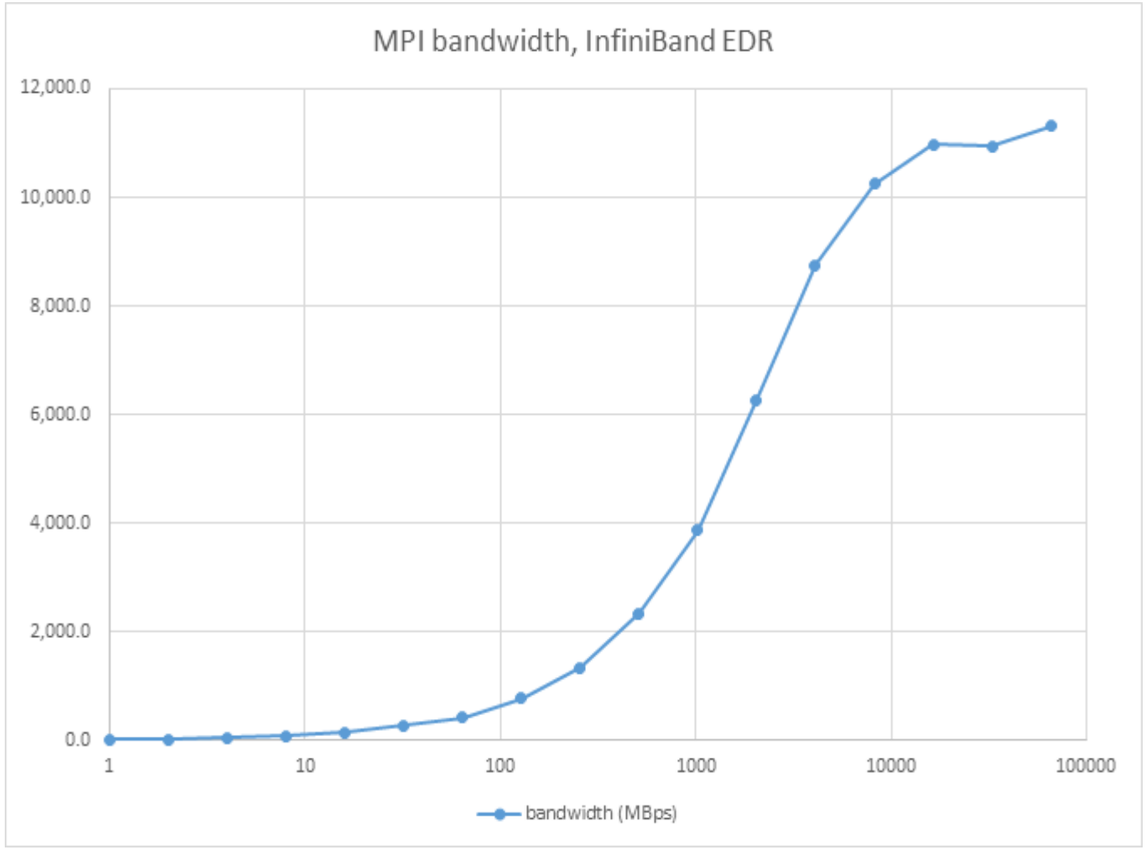
UCX Design

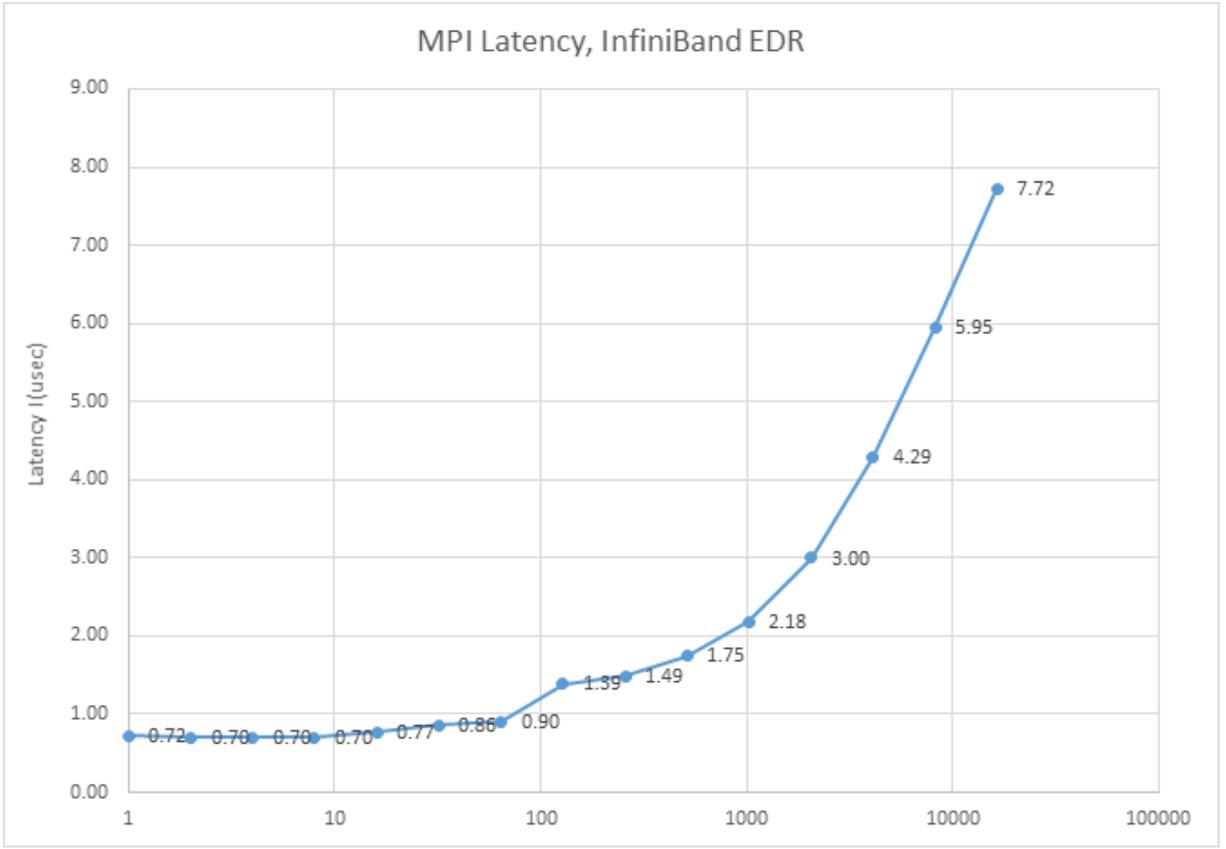

Applications
MPICH, Open-MPI, etc.
OpenSHMEM, UPC, CAF, X10, Chapel, etc.
Parsec, OCR, Legions, etc.
Burst buffer, ADIOS, etc.
UCX
UC-P (Protocols) - High Level API
Transport selection, cross-transrport multi-rail, fragmentation, operations not supported by hardware
Message Passing API Domain: tag matching, randevouze
PGAS API Domain: RMAs, Atomics
Task Based API Domain: Active Messages
I/O API Domain: Stream
UC-T (Hardware Transports) - Low Level API
RMA, Atomic, Tag-matching, Send/Recv, Active Message
Transport for InfiniBand VERBs driver
RC
UD
XRC
DCT
Transport for Gemini/Aries drivers
GNI
Transport for intra-node host memory communication
SYSV
POSIX
KNEM
CMA
XPMEM
Transport for Accelerator Memory communication
GPU
UC-S (Services)
Common utilities
Utilities
Data structures
Memory Management
OFA Verbs Driver
Cray Driver
OS Kernel
Cuda
Hardware

Mellanox’s UCX collaboration contributions

- Mellanox is moving towards using UCX as its vehicle for delivering InfiniBand HPC point-to-point communication capabilities.
- Co-maintainer of UCX.
- Is an active participant in defining UCX architecture and design.
- Provides and maintains InfiniBand support for UCX, that is optimized for current and next-generation hardware.
- Takes part in defining and implementing the upper-layer protocols.
- Maintains correctness and performance tests to ensure production quality.

Measured MPI performance with UCX

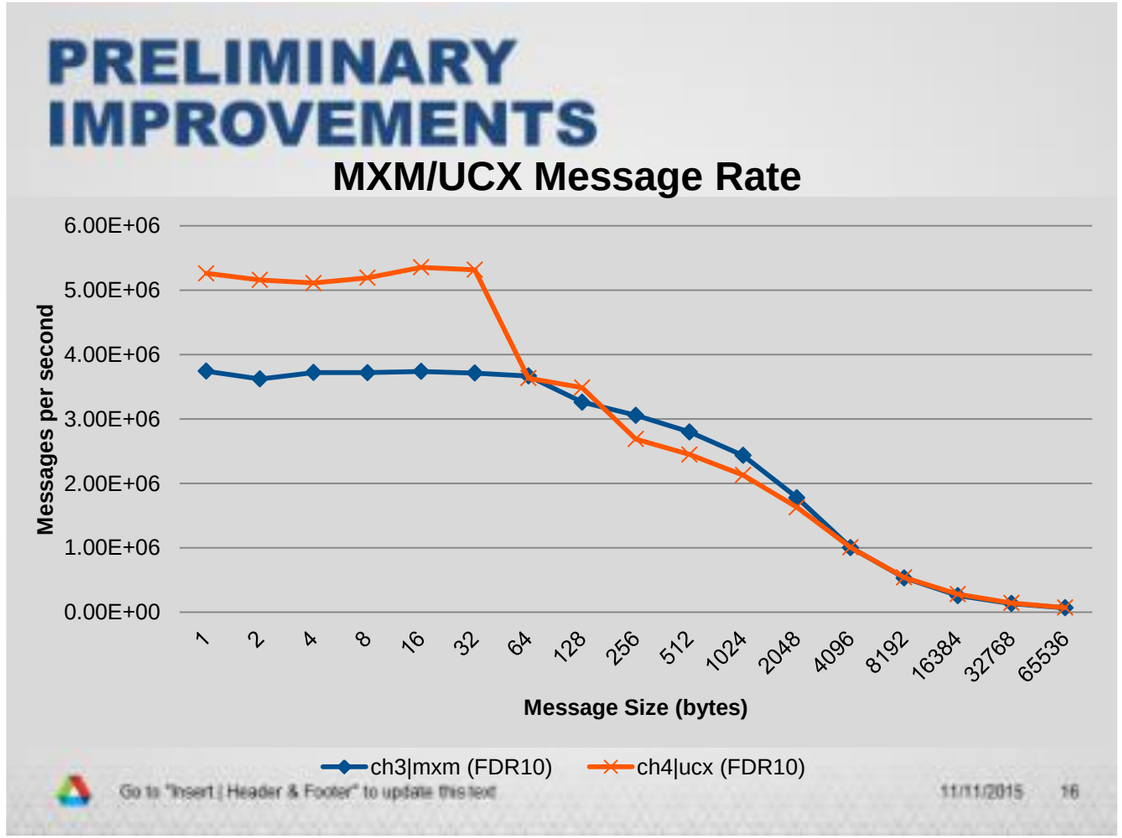




Mellanox MPICH CH4 Plans

- Highly supportive of the CH4 development effort. We are impressed with the preliminary results showing significant reduction in software overheads when compared against CH3. Great work, Argonne!

- We’re so impressed, in fact, we want to help in the effort.
  - Mellanox is planning to participate in Argonne UCX/CH4 hackathons and code camps.



Thank You!

Thank You

Mellanox TECHNOLOGIES

Connect. Accelerate. Outperform.™

MS-MPI

SC|15 Update

Microsoft is invested in HPC

- Microsoft is, and will continue, investing in HPC
  - All on-premises (HPC Pack evolution, SMB Direct)
  - Hybrid (burst via HPC Pack)
  - All in the cloud (Head Node in IaaS VM)
- RDMA support for both Windows and Linux Azure HPC VM
- We aim to provide the best commercial platform for compute-intensive parallel workloads

Committed to supporting the ecosystem

- MS-MPI is supported in all Windows environments:
  - Windows 10, 8.1, 8, 7
  - Windows Server (2008, 2008 R2, 2012, 2012R2)
  - Azure PaaS
  - Azure Batch
- Free to use
- Free to redistribute with 3rd party applications

Customer driven

- MS-MPI v6:
  - MPI_THREAD_MULTIPLE support
  - Initial support for nonblocking collectives
  - Matched Probe, Large Count, MPI-3 datatypes
- MS-MPI v7:
  - https://www.microsoft.com/en-us/download/details.aspx?id=49926
  - Almost complete support for nonblocking collectives
  - RPC-based PMI
  - MS-MPI launch service
- MS-MPI v.next:
  - Finish nonblocking collective support
  - MPI Tools Interface
  - Better developer support

We won’t bite!

- Let us know what you need
  - Features
  - Bugs
  - Help
- MS-MPI team eager to connect with Windows HPC community
  - Home page (http://msdn.microsoft.com/en-us/library/bb524831.aspx)
  - Contact askmpi@microsoft.com
  - Blog (http://blogs.technet.com/b/windowshpc/)
  - Web Forum (http://social.microsoft.com/Forums/en-US/home?forum=windowshpcmpi)

MPICH on K, post T2K, and post K and CH4

Masamichi Takagi,Norio Yamaguchi,
Masayuki Hatanaka, Yutaka Ishikawa,
RIKEN AICS

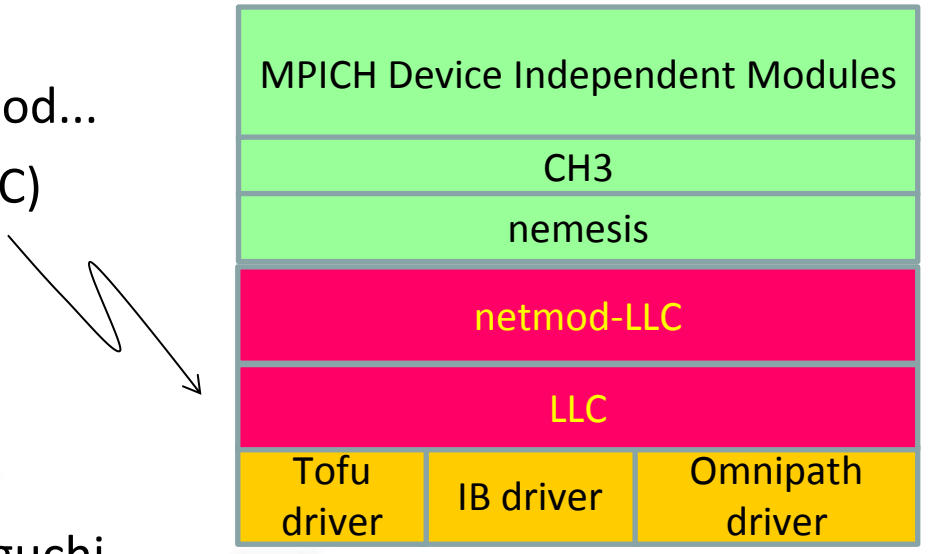
MPICH BoF at SC15
2015/11/17

RIKEN AICS

MPICH over Tofu and IB

Collaboration with MPICH team

- Developing MPICH/Tofu and MPICH/IB via netmod...
- On top of Low-Level Communication Library (LLC)



Developers

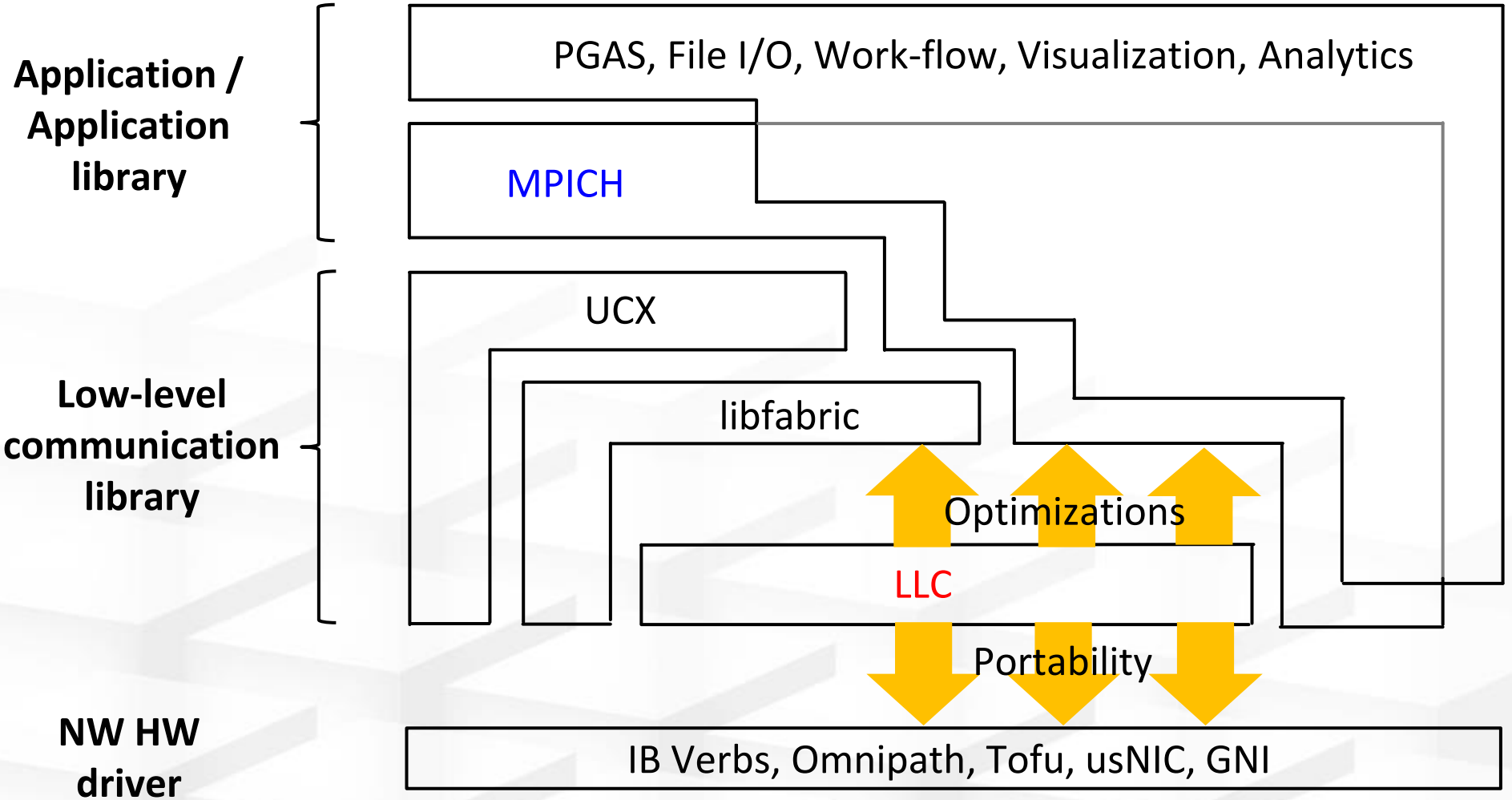
- MPICH for IB: Masamichi Takagi and Norio Yamaguchi
- MPICH for Tofu: Masayuki Hatanaka

Deployment target

- K and post K, Japanese supercomputers at RIKEN AICS
- Post T2K, manycore-based cluster operated in 2016 at Univ. of Tsukuba and Univ. of Tokyo

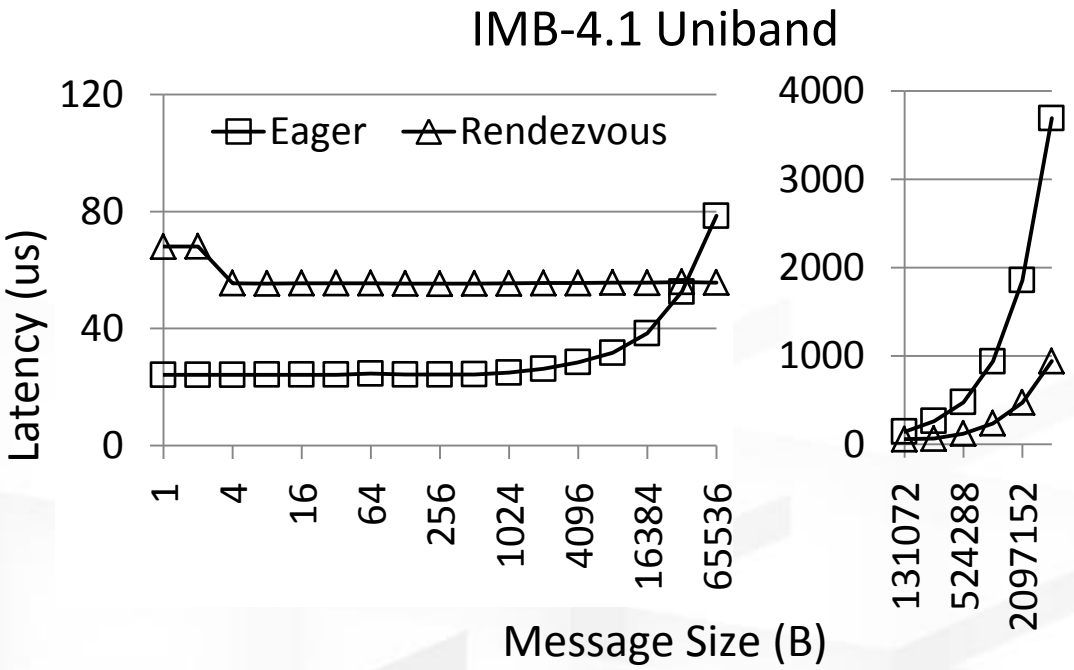
LLC

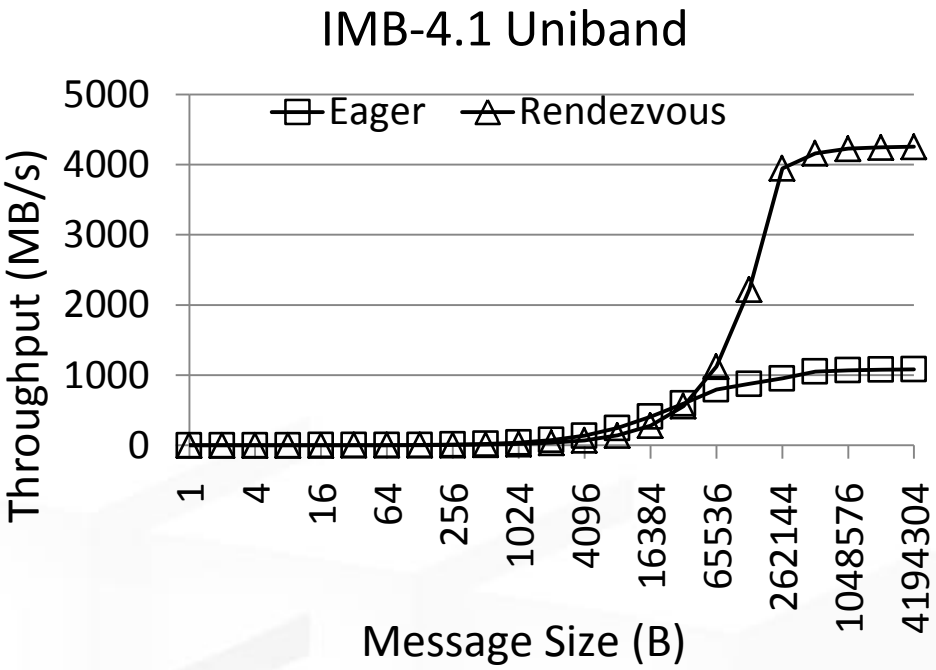
- Provide application-specific or site-specific optimizations to many app/libs
- Run on different NICs/HCAs



Current Status

- MPICH over LLC/Tofu
  - MPICH-3.1.4 with LLC-0.9.4 is available to K users as of Oct 1, 2015





- MPICH over LLC/IB
  - API specification is finalized in Q1 of 2015
  - Working on MPICH-3.2b3 and LLC-1.0 will be released at the end of Dec, 2015

Moving forward to CH4

Development

- Get involved in the design by sharing effort on CH3/LLC
- Planning hackathon to get the initial version of the LLC netmod for CH4

Deployment plan

- Post K: Planning to deploy CH4/LLC at launch, supported by AICS
- Post T2K: Planning to deploy CH3/LLC at launch, deploy CH4/LLC as an update

ParTec Cluster Competence Center

MANAGED HPC

ParaStation MPI

MPICH BoF · SC15 · Austin, TX
Nov 17, 2015

Norbert Eicker
Technical Lead – ParaStation Consortium

ParaStation Cluster Suite

- ParaStation ClusterTools
  - *Provisioning and Management*
- ParaStation HealthChecker & TicketSuite
  - *Automated error-detection & handling*
  - *Ensuring integrity of the computing environment*
  - *Error prediction*
  - *Keeping track of issues*
  - *Powerful analysis tools*

maximize job throughput
minimize administration

- ParaStation MPI & Process Management
  - *Run-time environment specifically tuned for the largest distributed memory supercomputers*
  - *Scalable & mature software setup*
  - *Batch System Integration (Torque, SLURM)*

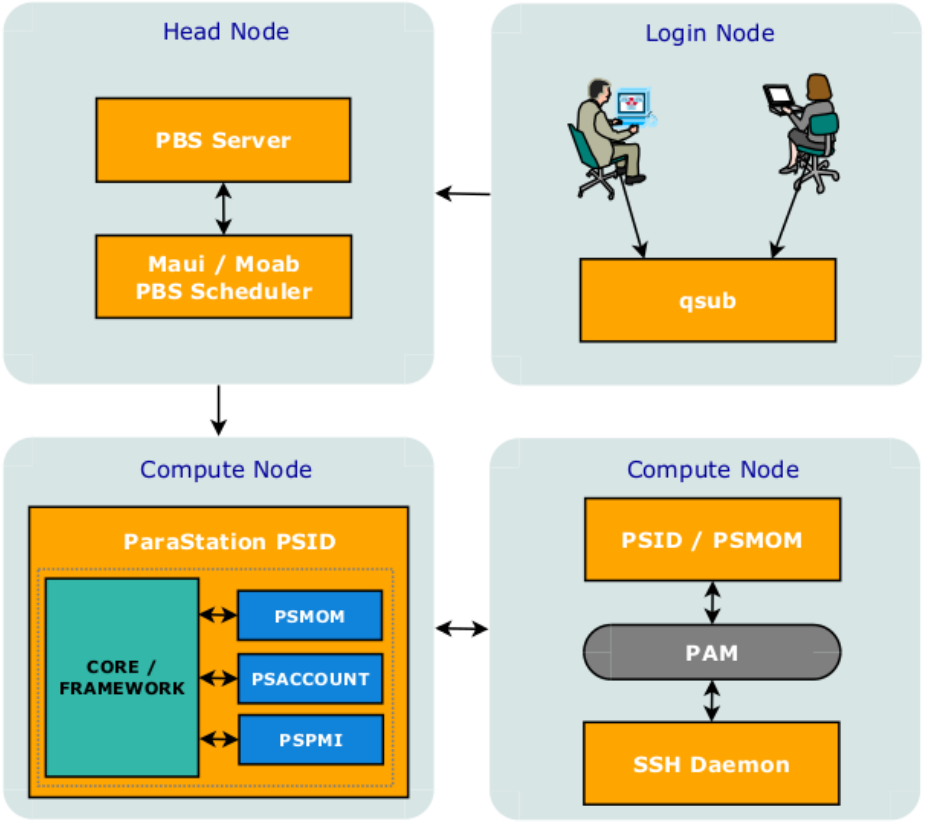
∑ ParaStation Cluster Suite

ParaStation MPI

- Implemented as an MPICH ADI3 device
- Powered by “pscom”: efficient, low-level communication library
- pscom supports a wide range of interconnects, even in parallel:
  - *TCP/IP*
  - *EXTOLL*
  - *DEEP: Cluster Booster Communication protocol*
  - *Shared Memory (optimized e.g. on Intel MIC)*
  - PSM (QLogic IB)
  - P4sock (GbE, 10GbE)
  - Verbs (IB)
  - DAPL (10GbE, IB)
- „Best“ interconnect negotiated at application start
- Dynamic loading of 3rd party communication libraries
  - → *One executable supports all interconnects*
  - → *3rd party libraries not needed while compiling / linking*
- Proven to scale up to 3,000 nodes and 39,360 processes per job

ParaStation Management Daemon

- Process startup and control facility
  - *User environment setup*
  - *Process pinning*
  - *Signal handling & I/O forwarding*
  - *Proper cleanup after jobs*
- Efficient and scalable comm. subsystem for inter-daemon messages (RDP based)
- Replaces the RMSs' execution daemons (Torque, SLURM)
  - *Reduces the no. of daemons*
  - *Enforces resource limits*
  - *Provides more precise monitoring*
  - *Offers optional extra features, like SSH login to allocated compute nodes during job run*
- Extensible via a plugin system



ParaStation MPI

- Supports all MPICH tools (e.g. for tracing and debugging)
- MPI libraries for various compilers, esp. GCC, Intel
- MPI management daemon on computational node has full knowledge of the jobs status
  - *Precise resource monitoring*
  - *Proper process cleanup*
  - *Interfaces to RMS (full integration with TORQUE and SLURM)*
  - *Efficient inter-daemon communication subsystem*
  - *No overbooking of nodes even without RMS*
- Start mechanism (mpiexec) also supports several other MPI libraries

ParaStation MPI

- ParaStation MPI is OpenSource

  → *available on https://github.com/ParaStation*

ParaStation MPI: Recent Developments

- MPI-3 features
  - *Extended Tool Support / Information Interfaces* ✓
    → `MPI_T` *Performance & Config. API*
  - *Extension of Collective Operations* ✓
    → `MPI_Neighborhood_()` *Family*
  - *Non-blocking Collectives* ✓
    → `MPI_Ibcast()` *& Co.*
  - *Support for Hybrid Programming* ✓
    → `MPI_Mprobe()` / `MPI_Mrecv()`
  - *Extensions for One-Sided Communication* ✓
    → *e.g. RMA for "real" Shared-Memory*
    `MPI_Win_allocate_shared()`

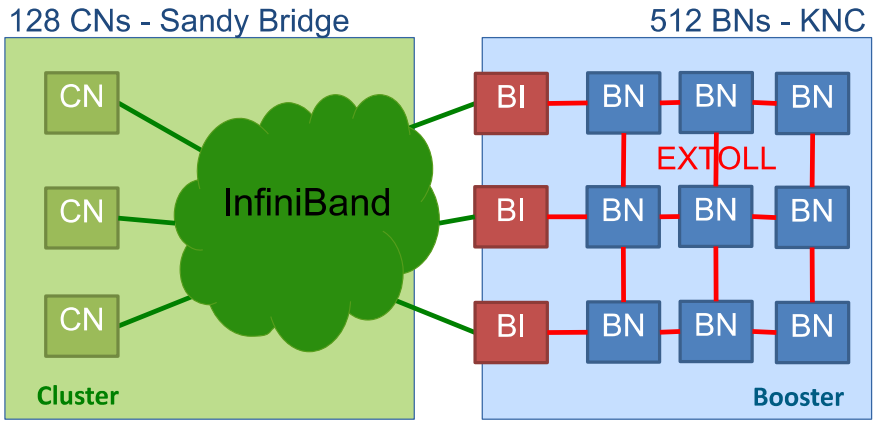
ParaStation MPI

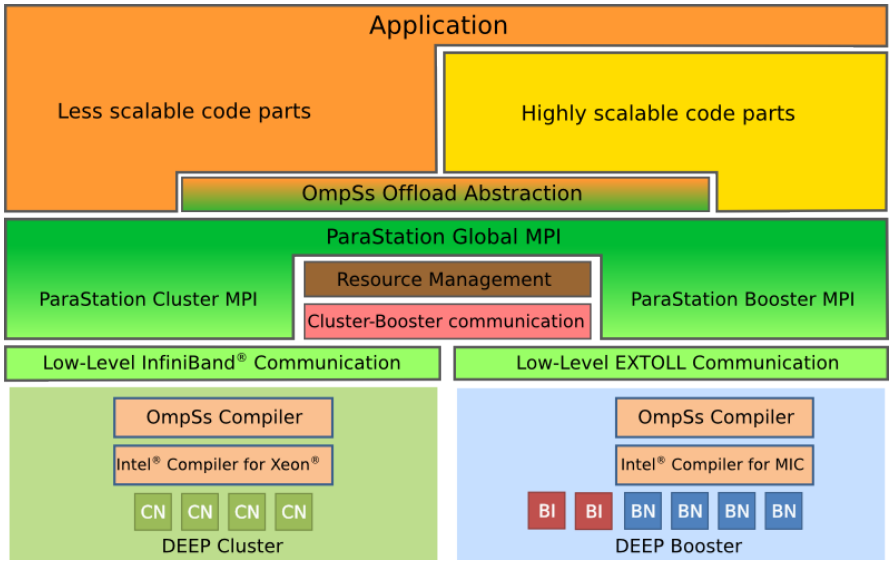
- FaST:
  Find a Suitable Topology for Exascale
  - *Support for application-transparent VM migration*
  - *Shutdown/Reconnect support for pscom library*
  - *Working prototype with little to no runtime overhead, and InfiniBand support*
  - *Joint Research Project funded by German Federal Ministry of Education and Research.*

- ParaStation MPI in the DEEP-Project
  - *See next slide*
  - *This project has received funding from the European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement nr. 287530 (DEEP).*

ParaStation Global MPI in DEEP

- Heterogeneous system: Cluster-Booster architecture
  - *Cluster Nodes (CN) with Intel Xeon multi-core CPUs*
  - *Booster Nodes (BN) with Intel MIC many-core CPUs*
  - *Booster Interfaces (BI) connecting Cluster and Booster*





- ParaStation MPI powers DEEP's MPI-based offloading mechanism
  - *Using inter-communicators based on MPI_Comm_spawn()*
  - *Offloading of highly-scalable code parts to the Booster*
  - *Enable transparent Cluster-Booster data exchange via MPI*

Reference Installations at JSC

JuRoPA

- 308 TFlop/s peak
- #10 of Top500 list (June 2009)
- Production 8/2009 - 6/2015
- 3288 compute nodes
  (Sun SB6048 / Bull NovaScale R422-E2)
- Intel Xeon X5570 (Nehalem) / 24 GB
- Mellanox QDR
- ParaStation Software Stack / Torque

JURECA

- 1.8 (CPU) / 0.44 (GPU) PFlop/s peak
- #50 of Top500 list (Nov 2015)
- Production started 10/2015
- 1872 compute + 64 fat nodes
  (T-Platforms V-Class / SuperMicro)
- Intel Xeon E5-2680v3 (Haswell) / 128+ GB
- Mellanox EDR
- ParaStation Software Stack / SLURM

ParTec enables HPC

- Strong general purpose cluster specialist for more than a decade
  - Spin-off of the University of Karlsruhe
  - *Working as SME since 1999 in the fields of cluster computing*
- Unrivaled expertise in developing Cluster Software
- ParTec was elected as the partner of choice in some leading HPC sites across Europe
- Engagement and active development in projects towards Exascale supercomputers

- Technical activities
  - *JuRoPA (10th of Top500 in 06/2009)*
  - *JURECA (50th of Top500 in 11/2015)*
  - *ExaCluster Lab*
  - *DEEP Project*
  - *DEEP-ER Project*
  - *ParaStation Consortium*
- Political activities
  - *EOFS & Exascale10*
  - *PROSPECT e.V.*
  - *ETP4HPC*

Thank you!

Questions?

http://www.par-tec.com

MPICH on TH-Express Interconnect

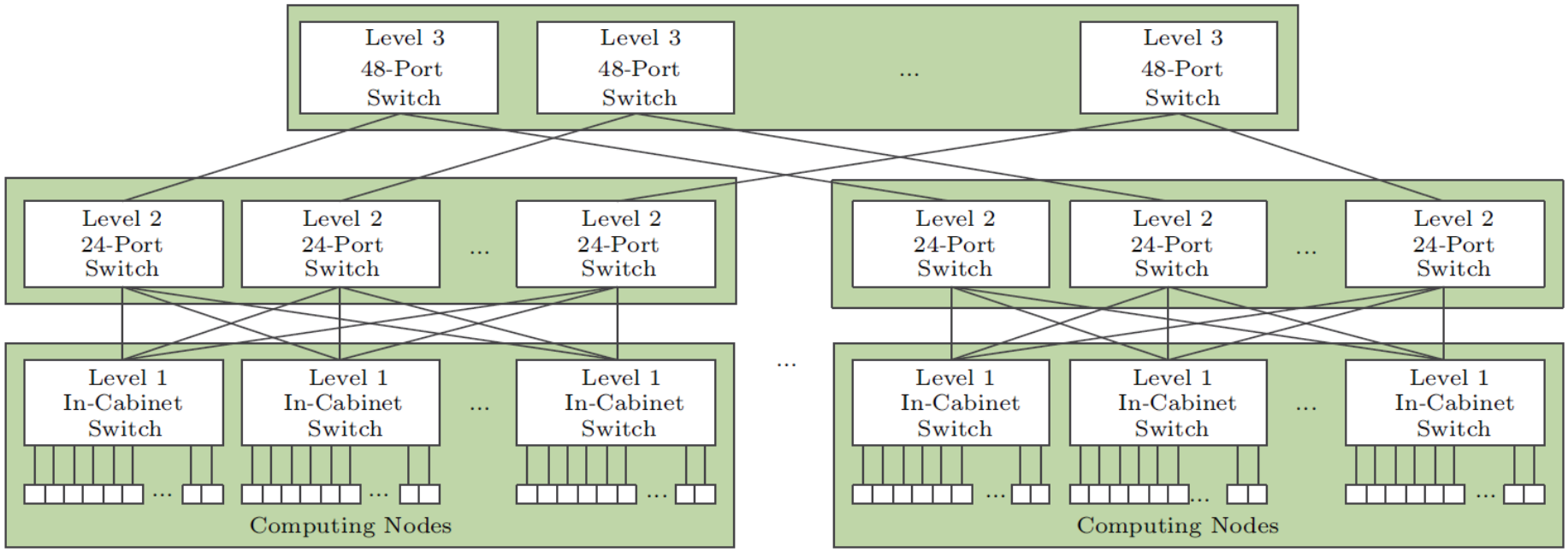
Yanhuang Jiang

School of Computer Science
National University of Defense Technology

SC'15 MPICH BOF

TH-Express Interconnect

- High bandwidth network port
  - 8Lane@14Gbps
- Fat-tree topology, and adaptive routing method



TH-Express Interconnect

- Network Interface
  - virtual ports with Mini-Packet and RDMA operations for user-level communication
  - Programming I/O for low latency and high rate message passing
  - Triggered operations for communication offload
  - End-to-end communication reliability

MPICH on TH-Express: current status

- Based on MPICH 3.1.4
- MPI-3 Compliant
- Implemented as a Nemesis netmod: glex
  - It can be integrated into new MPICH version quickly
  - construct correct message order from out of order network packets
  - data transfer protocol based on RDMA
  - exclusive RDMA resource for performance
  - shared RDMA resource with dynamic flow control for scalability
  - zero-copy long message transfer using RDMA and memory registration cache

MPICH on TH-Express: current status

- Collective offload
  - Algorithms based on k-nomial or k-ary tree using NIC triggered operations
  - Offloaded implementation of MPI_Barrier, MPI_Bcast, MPI_reduce/MPI_Allreduce, and several non-blocking collective interfaces
- MIC symmetric mode
  - Implemented in GLEX-direct, similar to Intel PSM-Direct for MIC
  - Thus, MPICH-TH is just recompiled without modification for symmetric mode MIC MPI processes

MPICH on TH-Express: current status

- Scalable processes startup with a customized PMI implementation as Slurm plugin
  - Hierarchical communication topology between process managers
  - KVS data cache in node, only changed KVS data being transferred

What's next

- Move to MPICH 3.2.X
  - More optimization on asynchronous progress
    - improving the overlap of computation and communication is really helpful for many applications
  - RMA optimization using 3.2.x framework
  - Offloaded optimization on more collective operations

Thanks

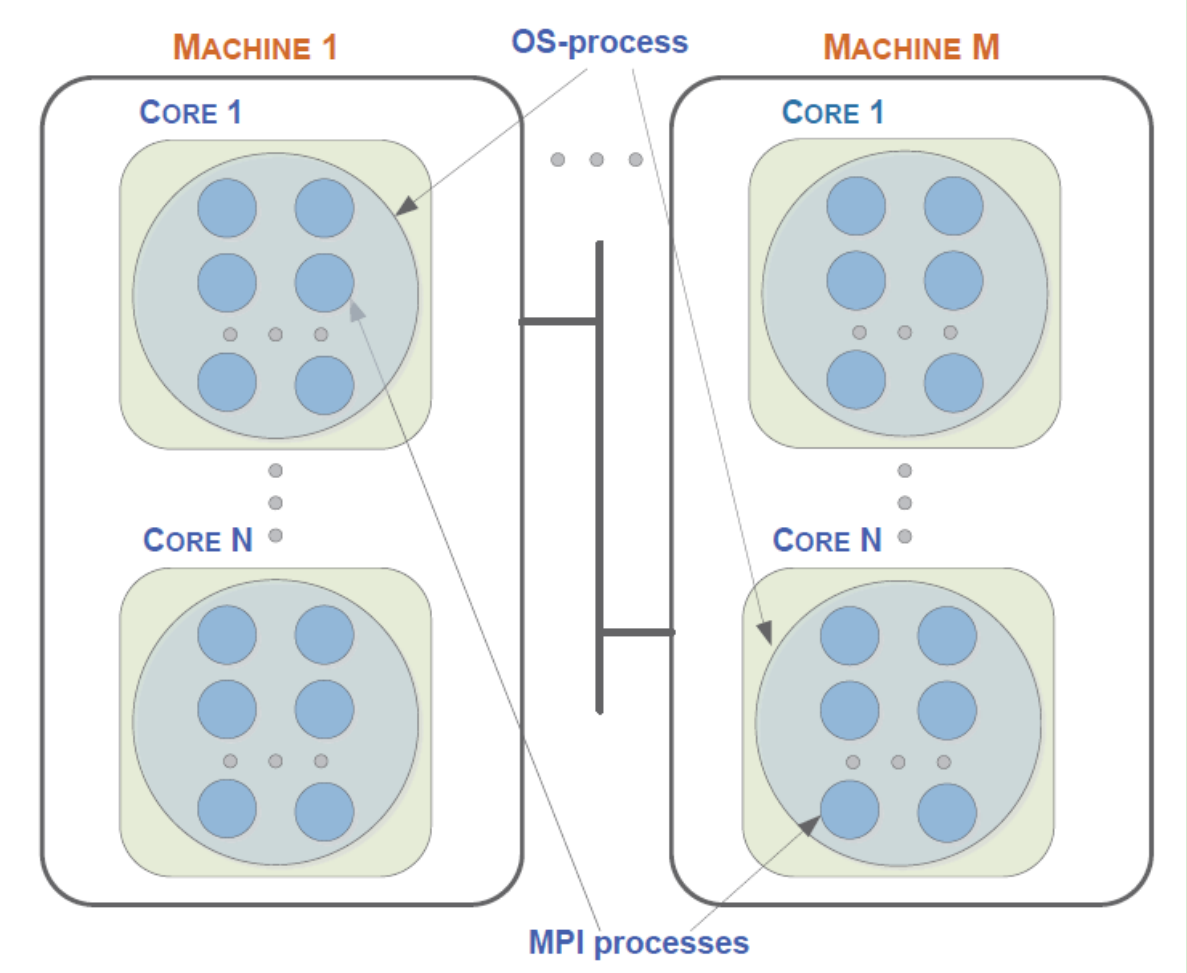
Fine-Grain MPI

Humaira Kamal and Alan Wagner
Department of Computer Science
University of British Columbia

FG-MPI extends the execution model of the Message Passing Interface (MPI) to expose **large-scale, fine-grain concurrency**.

FG-MPI is integrated into the latest version of **MPICH**

Exposes fine-grain concurrency at different levels in the system.



Added concurrency is easy to express

```
mpiexec -nfg 1000 -n 4 myprog
```

```
mpiexec -nfg 500 -n 8 myprog
```

```
mpiexec -nfg 500 -n 3 myprog : -nfg 250 -n 2 myprog :
        -nfg 1000 -n 2 myprog
```

4000 MPI processes

Development features of FG-MPI

Develop MPI programs that scale to hundreds and thousands on their notebooks and workstations.

Flexibility to **select different runtime schedulers** on the command line.

Use a deterministic process scheduler (e.g. round robin) **to debug and test** programs.

Run all MPI processes inside a single OS-process **to detect program safety issues** like deadlock.

Easy porting of many MPI programs to FG-MPI through the addition of a small bit of boiler-plate code.

Summary

- Has light-weight, scalable design integrated into MPICH middleware which leverages its architecture.
- Implements location-aware communication inside OS-processes and nodes.
- Allows the user to scale to millions of MPI processes without needing the corresponding number of processor cores.
- Allows granularity of MPI programs to be adjusted through the command-line to better fit the cache leading to improved performance.

Summary

- Enables design of novel algorithms and vary the number of MPI processes to match the problem rather than the hardware.
- Enables task oriented program design due to decoupling from hardware and support for function-level concurrency.
- Allows the programmer to focus on what needs to be scheduled rather than how to manage it.

FG-MPI is available
for download at:

http://www.cs.ubc.ca/~humaira/fgmpi.html

THANK-YOU

Thanks to 

support of this project

CH4 Next Steps

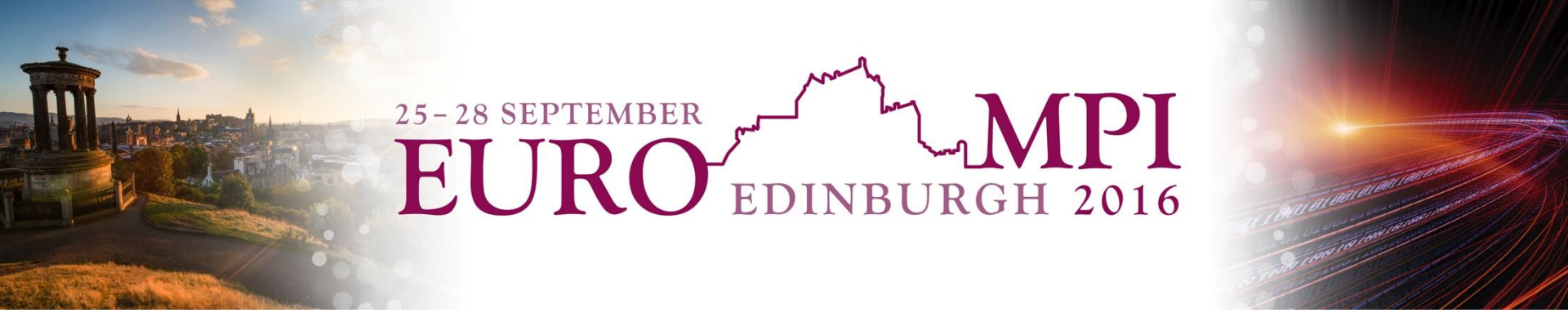
- Hackathons to kickstart development
- Weekly telecons starting December
  - Doodle poll to determine day and time

Argonne NATIONAL LABORATORY

Thanks for coming

See you at SC16 in Salt Lake!

U.S. DEPARTMENT OF ENERGY



www.EuroMPI2016.ed.ac.uk

- Call for papers open by end of November 2015
- Full paper submission deadline: 1st May 2016

- Associated events: tutorials, workshops, training
- Focusing on: benchmarks, tools, applications, parallel I/O, fault tolerance, hybrid MPI+X, and alternatives to MPI and reasons for not using MPI