Open MPI State of the Union XI Community Meeting SC17

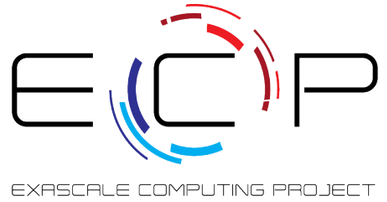
Jeff Squyres

George Bosilca

Josh Hursey

Gilles Gouaillardet

David Bernholdt



Interactive / Online / SC thingy

- Online question topic submission: Linklings
- BOF feedback form

www.open-mpi.org/sc17/

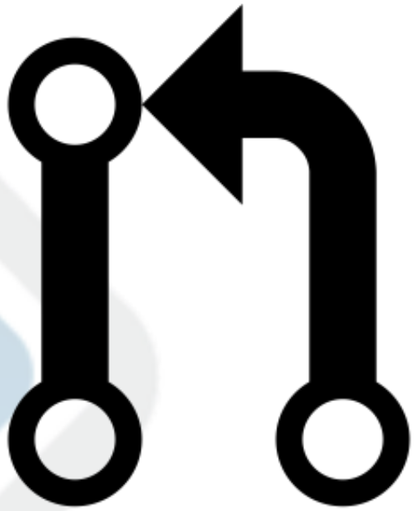
https://eurompi2018.bsc.es/

Full paper submission deadline: 1st May 2018

Quick Review

GitHub / Community Contributions

Contribute!

We ❤ 

(we love GitHub pull requests)

Contribution policy

- “Signed-off-by” required in commit messages:

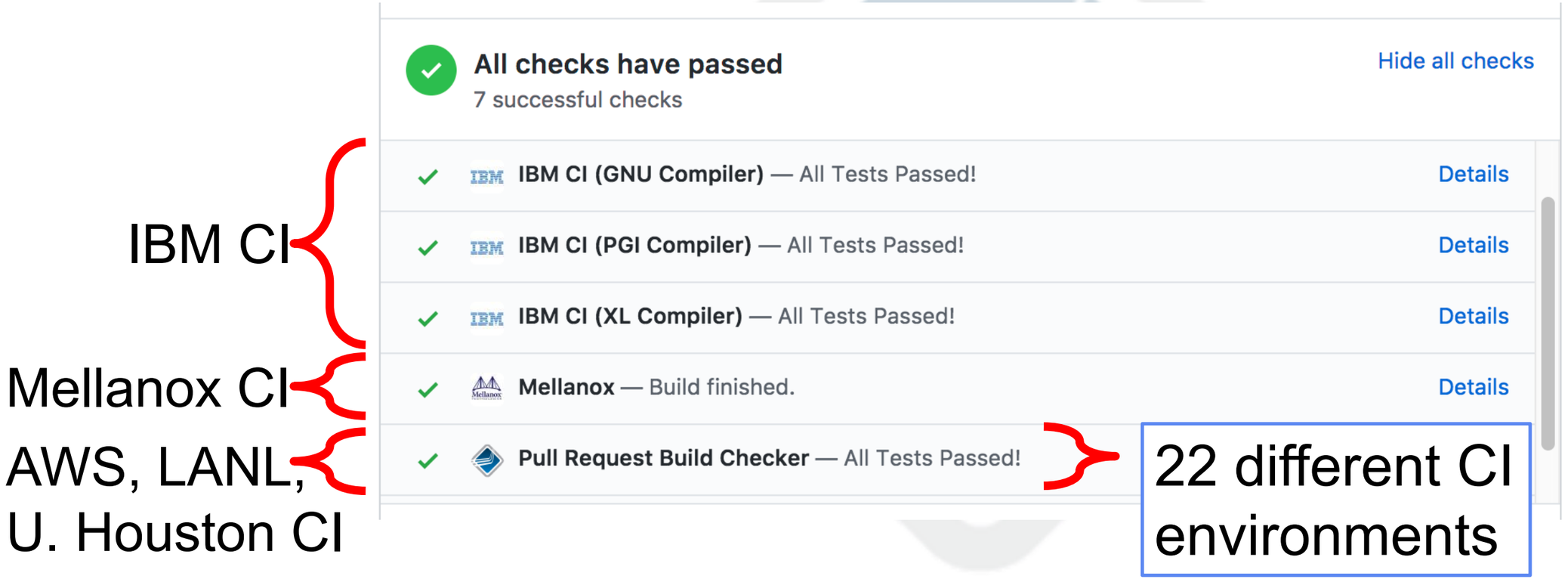
```
Some awesome new feature

Signed-off-by: Jeff Squyres <jsquyres@cisco.com>
```

- Open MPI Contributor’s Declaration
  - See the full definition here
  - Can automatically be added by “`git commit –s`”

Lots of CI and release engineering automation

- CI provided by community members
  - Special thanks to AWS for significant automation / CI investment



Open MPI versioning

Open MPI versioning

- Open MPI uses "A.B.C" version number triple
- Each number has a specific meaning:

A This number changes when backwards compatibility breaks

B This number changes when new features are added

C This number changes for all other releases

Definition

- Open MPI vY is *backwards compatible* with Open MPI vX (where Y>X) if:
  - Users can compile a correct MPI / OSHMEM program with vX
  - Run it with the same CLI options and MCA parameters using vX or vY
  - The job executes correctly

What does that encompass?

- “Backwards compatibility” covers several areas:
  - Binary compatibility, specifically the MPI / OSHMEM API ABI
  - MPI / OSHMEM run time system
  - `mpirun` / `oshrun` CLI options
  - MCA parameter names / values / meanings

What does that *not* encompass?

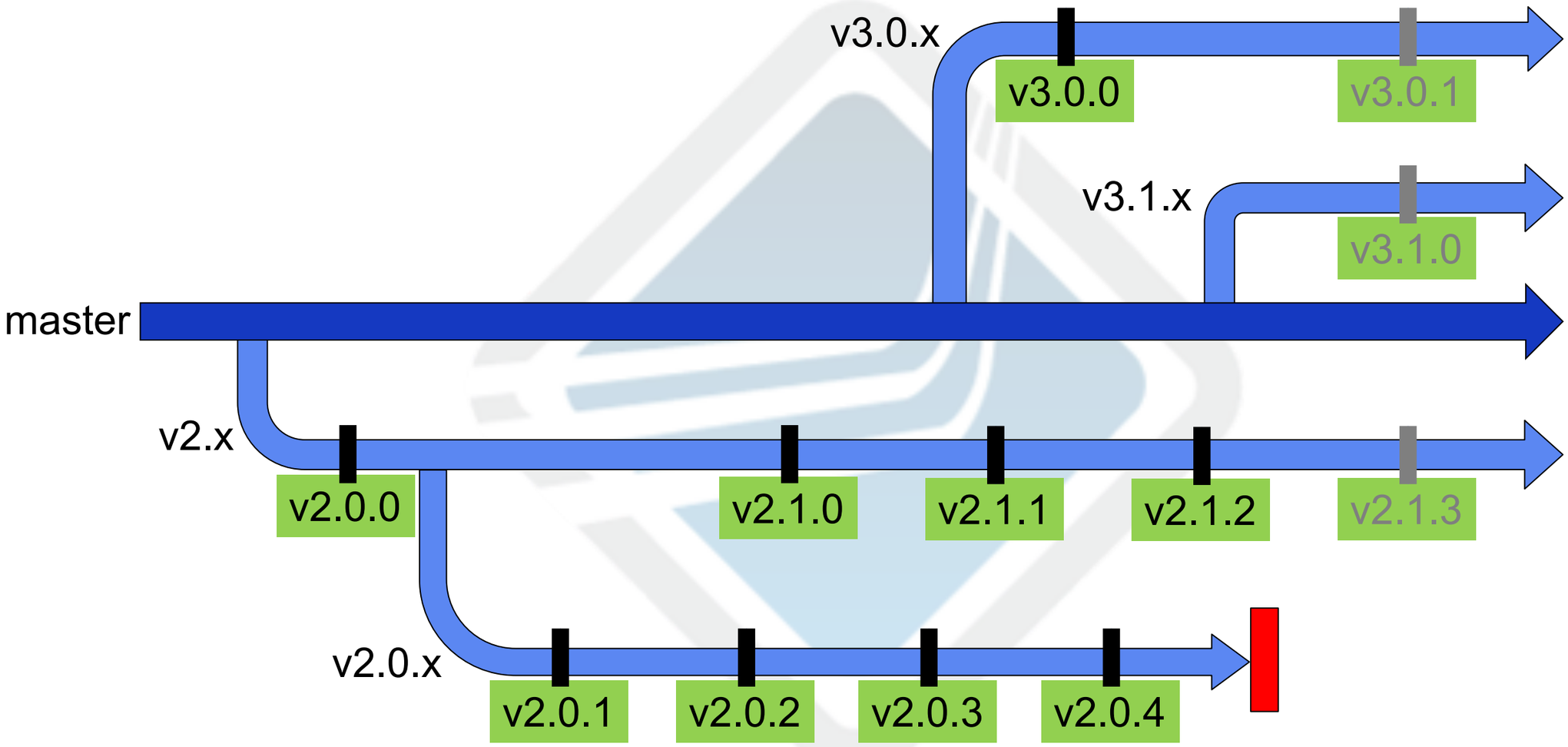
- Open MPI only supports running exactly the same version of the runtime and MPI / OSHMEM libraries in a single job
  - If you mix-n-match vX and vY in a single job…

…except for one (new) case
See PMIx slides, later

Current version series

- V2.0.x series
  - End of life
- v2.1.x series
  - Prior stable series
- v3.0.x series
  - Current stable series
- v3.1.x series
  - Upcoming series

Releases / branches



Version Roadmaps

v2.0.x series (EOL)

- Release managers
  - Howard Pritchard, Los Alamos National Lab
  - Jeff Squyres, Cisco Systems, Inc.

- Last release: v2.0.4
  - November 10, 2017
  - Minor / accumulated bug fixes
- End of life
  - No further releases expected

v2.1.x series (prior stable)

- Release managers
  - Howard Pritchard, Los Alamos National Lab
  - Jeff Squyres, Cisco Systems, Inc.

- Current release: v2.1.2
  - September 20, 2017
  - v2.1.3 expected Q1 2018
- Maintenance only
  - No new features
- Backwards compatible with v2.0.x
  - v2.0.x users (strongly) encouraged to upgrade

v3.0.x series (current stable)

- Release managers
  - Brian Barrett, AWS
  - Howard Pritchard, Los Alamos National Lab

- Current release: v3.0.0
  - September 12, 2017
  - v3.0.1 expected Q1 2018
- Maintenance only
  - No new features
- Not backwards compatible with v2.x
  - v2.x users encouraged to investigate / upgrade

v3.1.x series (upcoming)

- Release managers
  - Ralph Castain, Intel
  - Brian Barrett, AWS

- Expected release:
  - Q1 2018
- Minor new features
- Backwards compatible with v3.0.x

Deprecation notice: MPIR

- MPIR interface is used internally to launch / attach tools and debuggers
- The maintainer for Open MPI's MPIR is retiring!
- Unless someone else takes over, this is the plan:
  - Deprecation notice in NEWS in early CY2018
  - User runtime warnings in mid/late CY2018
  - Removal in CY2019

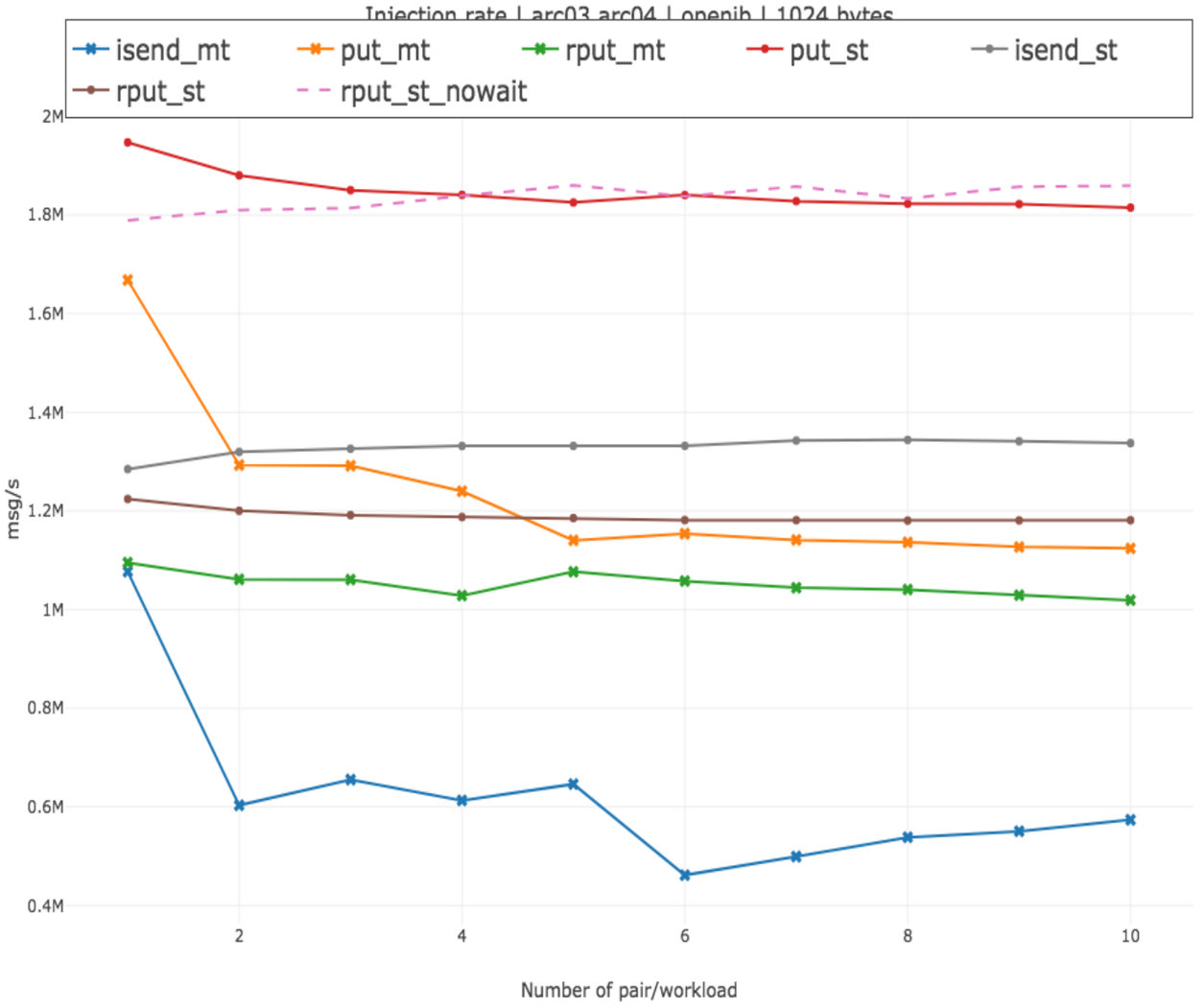
Threading/Collectives/Resilience/Tools

George Bosilca

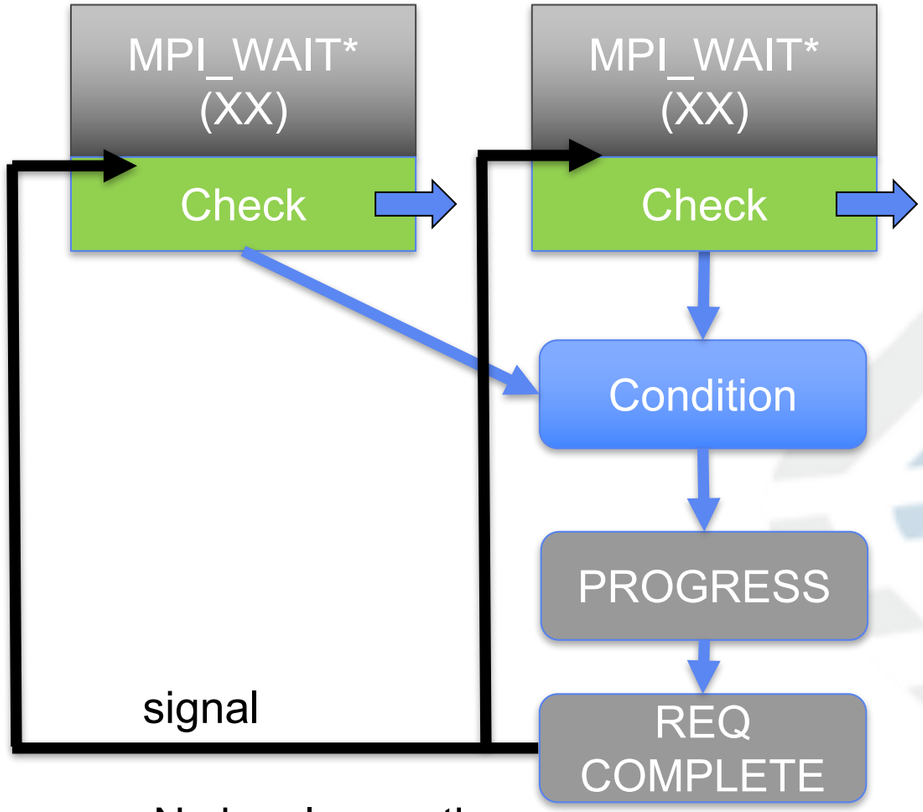
University of Tennessee

Injection rate in OMPI

- Decent performance for most of the one-sided approaches
  - Point-to-point less efficient
  - Multi-threaded …
- Problems:
  - The matching
    - Out-of-sequence messages
  - Out-of-order fragments in the PML
  - Request completion
  - Progress
- OMPI support multi-threading



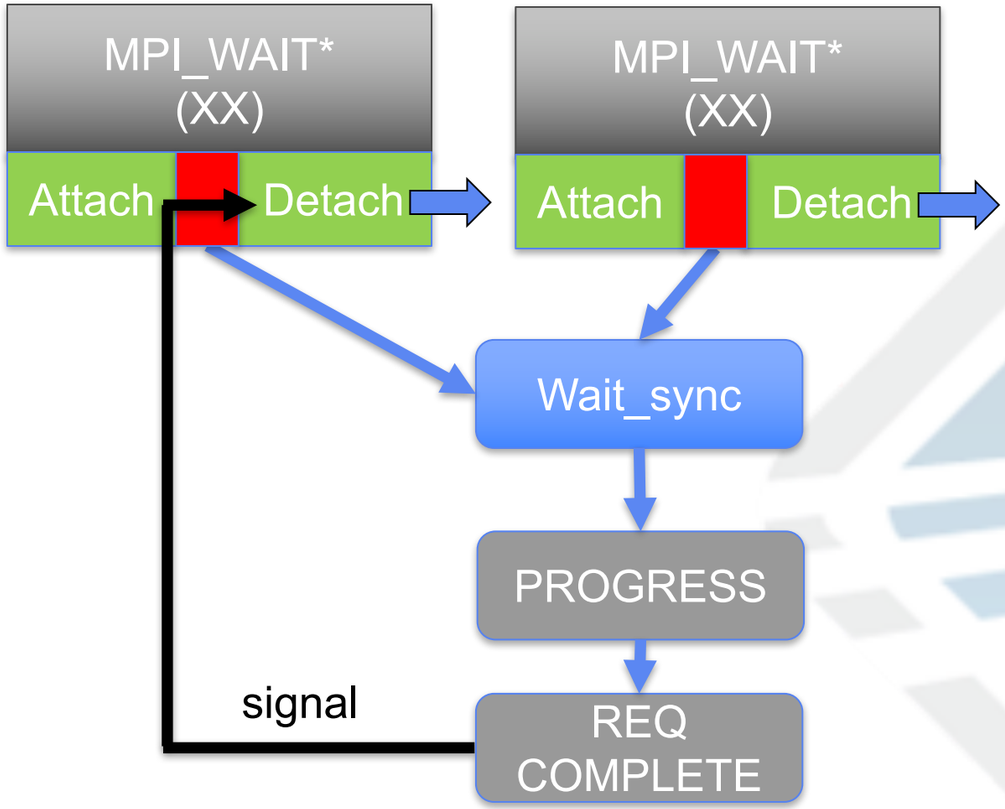
MPI_THREAD_MULTIPLE (Legacy design)



N signal operation
2N mutex operation
N progression

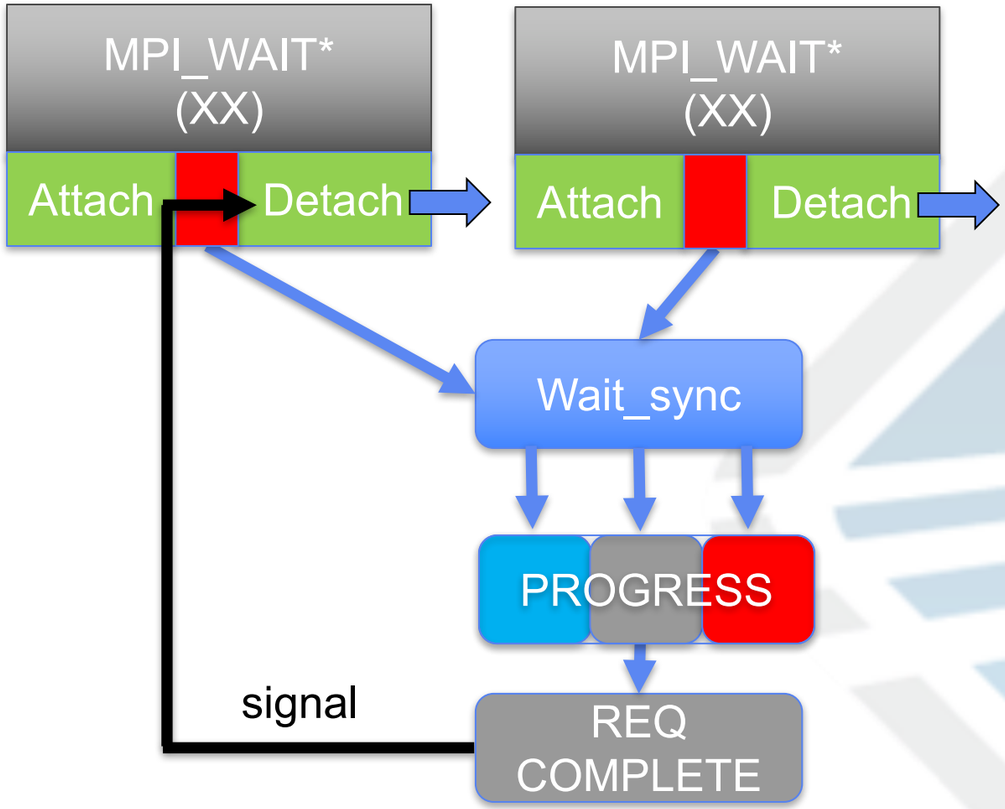
- Each request completion wakes up all threads in waiting mode
  - Cache (un)friedly: each thread has to check again the status of all associated requests
- Single thread in the progress at any moment
  - Network drained at the speed of a thread
  - No help from the other threads that are in the MPI library

MPI_THREAD_MULTIPLE (New design)



- Each request completion only affects the corresponding wait_sync synchronization object
- A thread only becomes active when the waiting condition become true (any, some, all)
  - Otherwise a thread sleeps, saving energy
- Single thread in the progress at any moment
  - Network drained at the speed of a thread
  - No help from the other threads that are in the MPI library

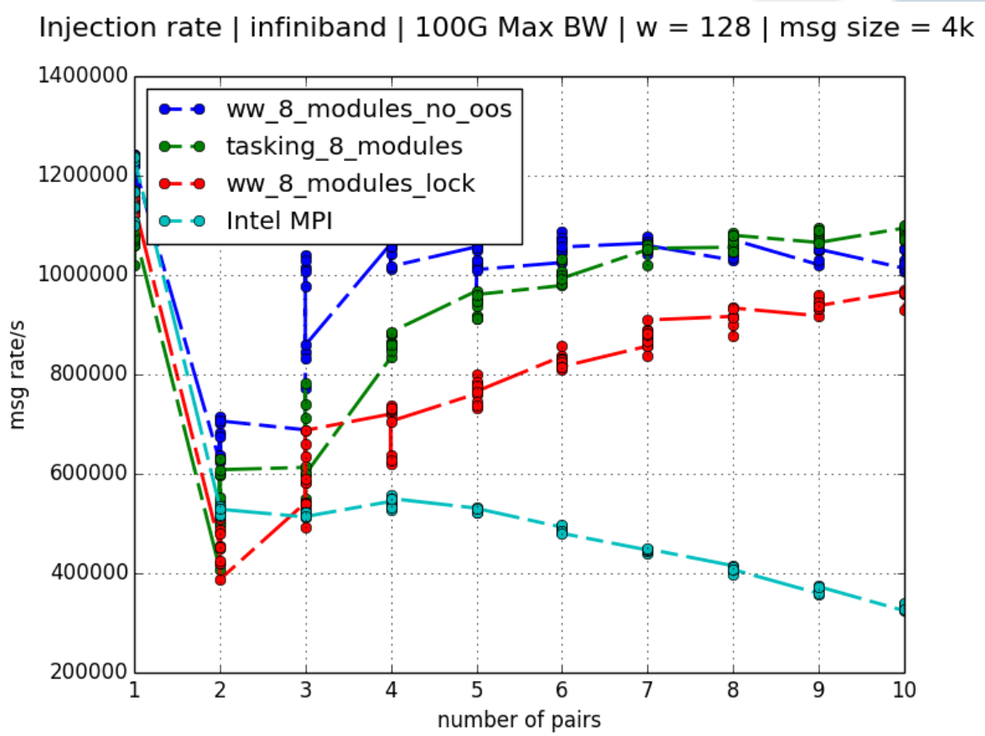
MPI_THREAD_MULTIPLE (New design)

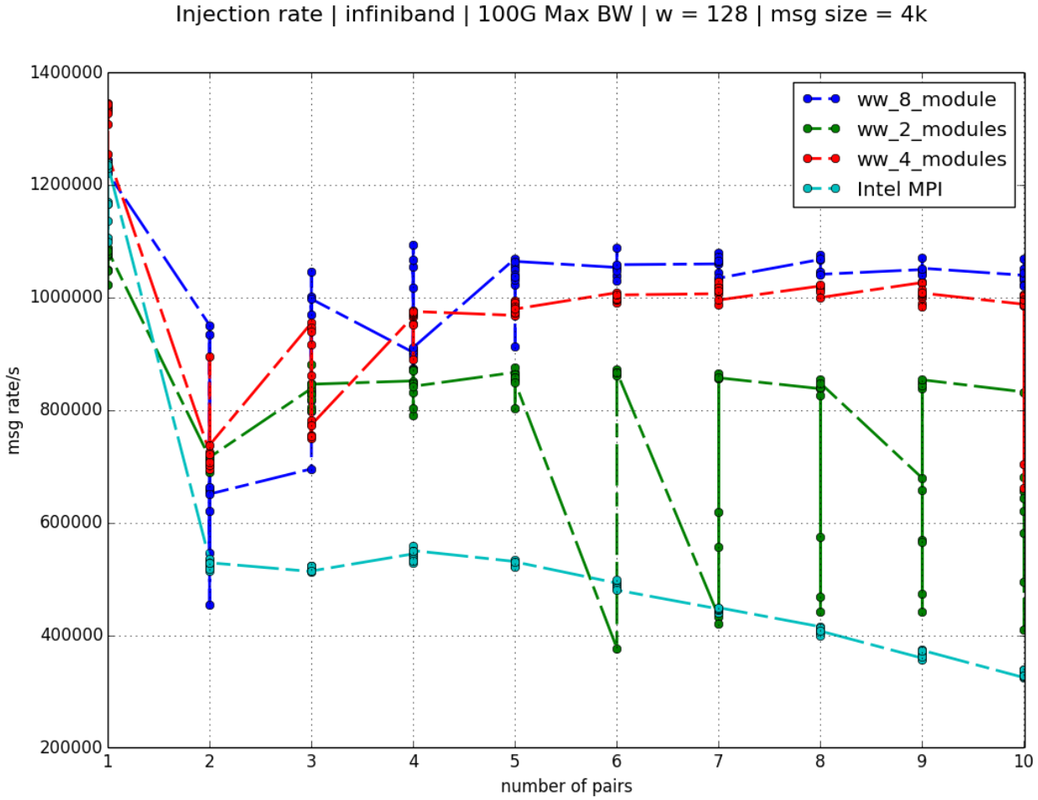


- Each request completion only affects the corresponding wait_sync synchronization object
- A thread only becomes active when the waiting condition become true (any, some, all)
  - A thread is now available to help
- Multiple threads in the progress
  - Network drained faster
  - Generic helper (progress BTLs)
  - Specialized helper (tasking based on the current needs)
    - Partial progress, collective op, packing/unpacking, …

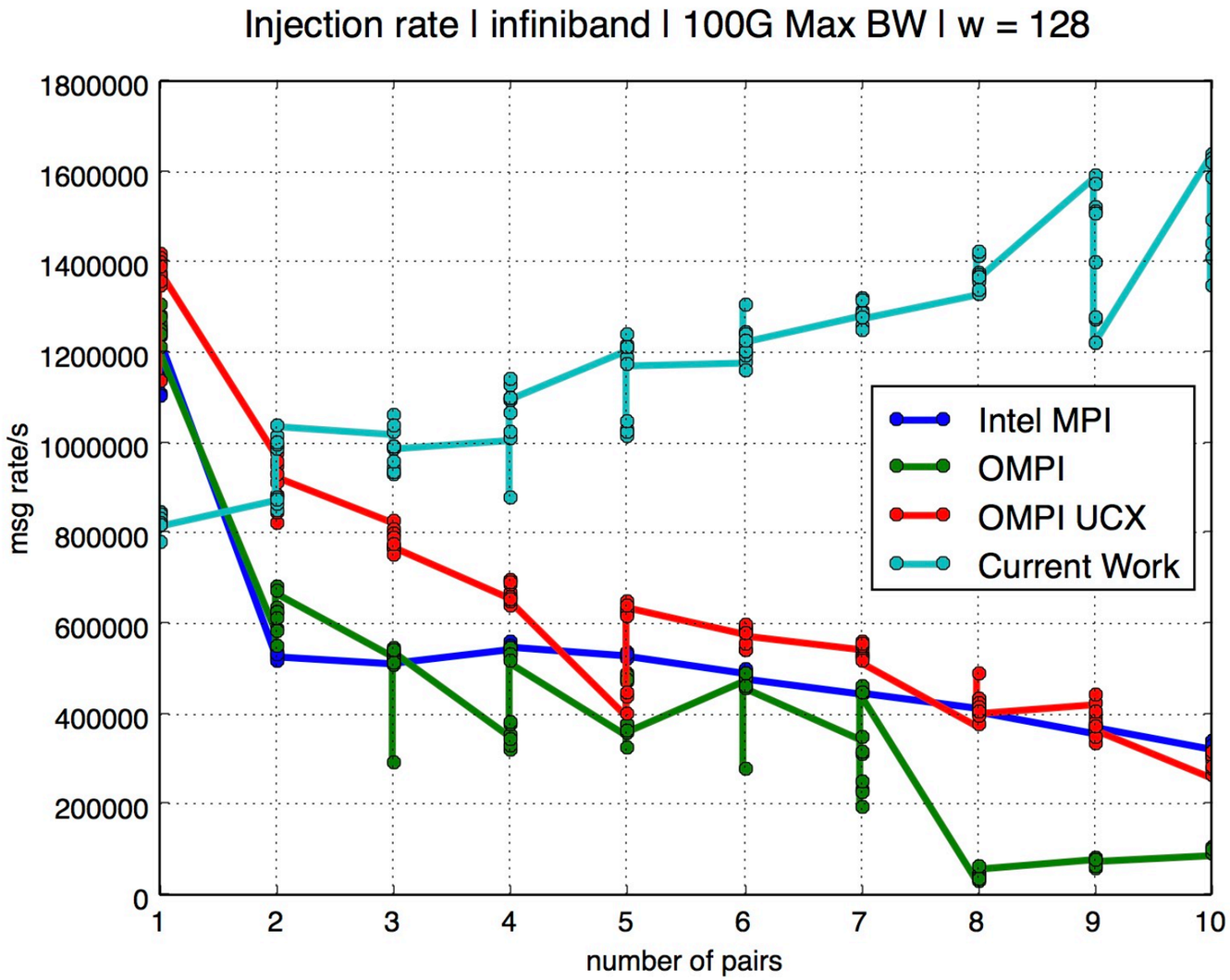
MPI_THREAD_MULTIPLE (New design)

- With these changes we moved the bottleneck
  - We need to increase the number of insertion/extraction channels





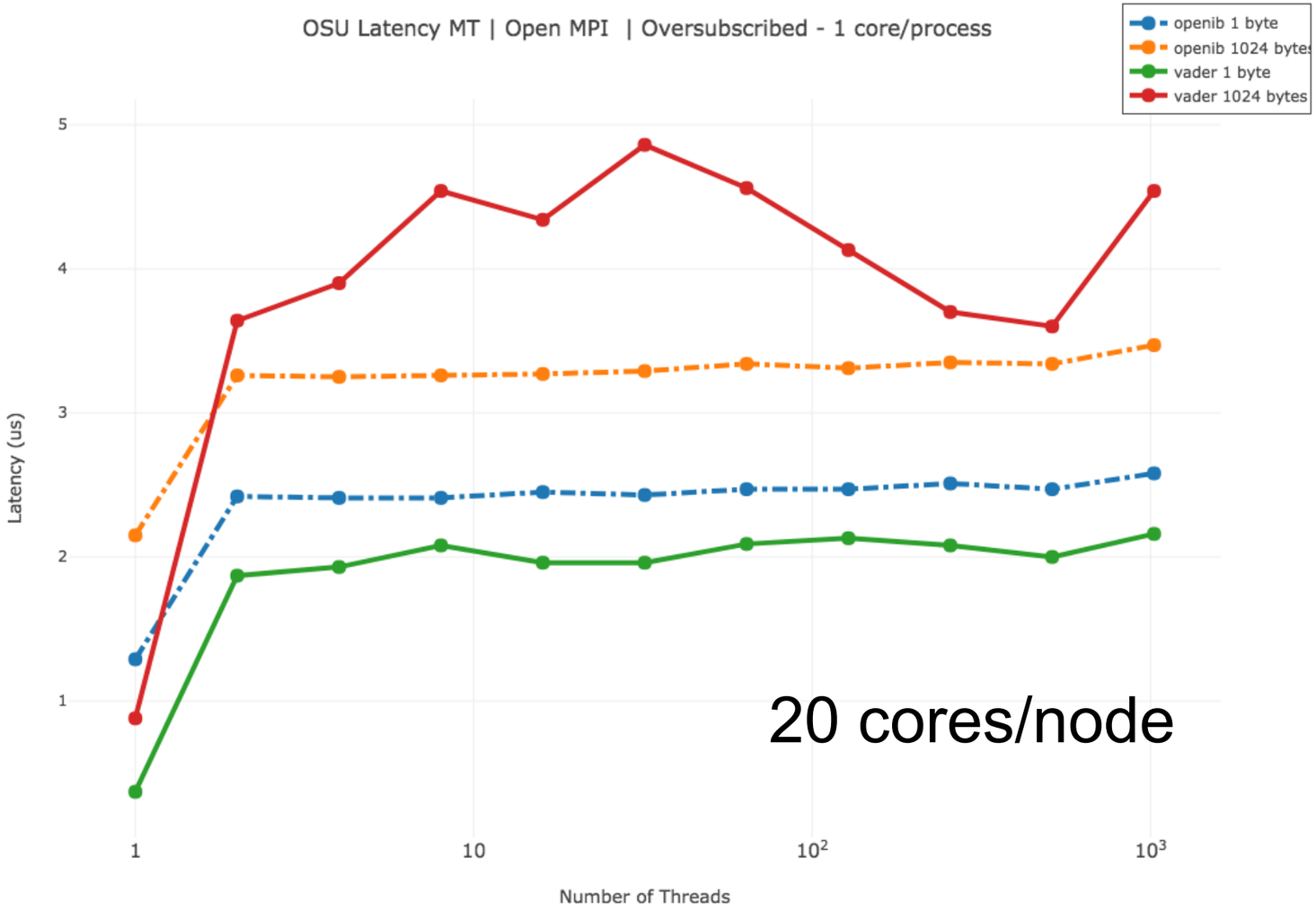
MPI_THREAD_MULTIPLE (New design)



- Multi-modules

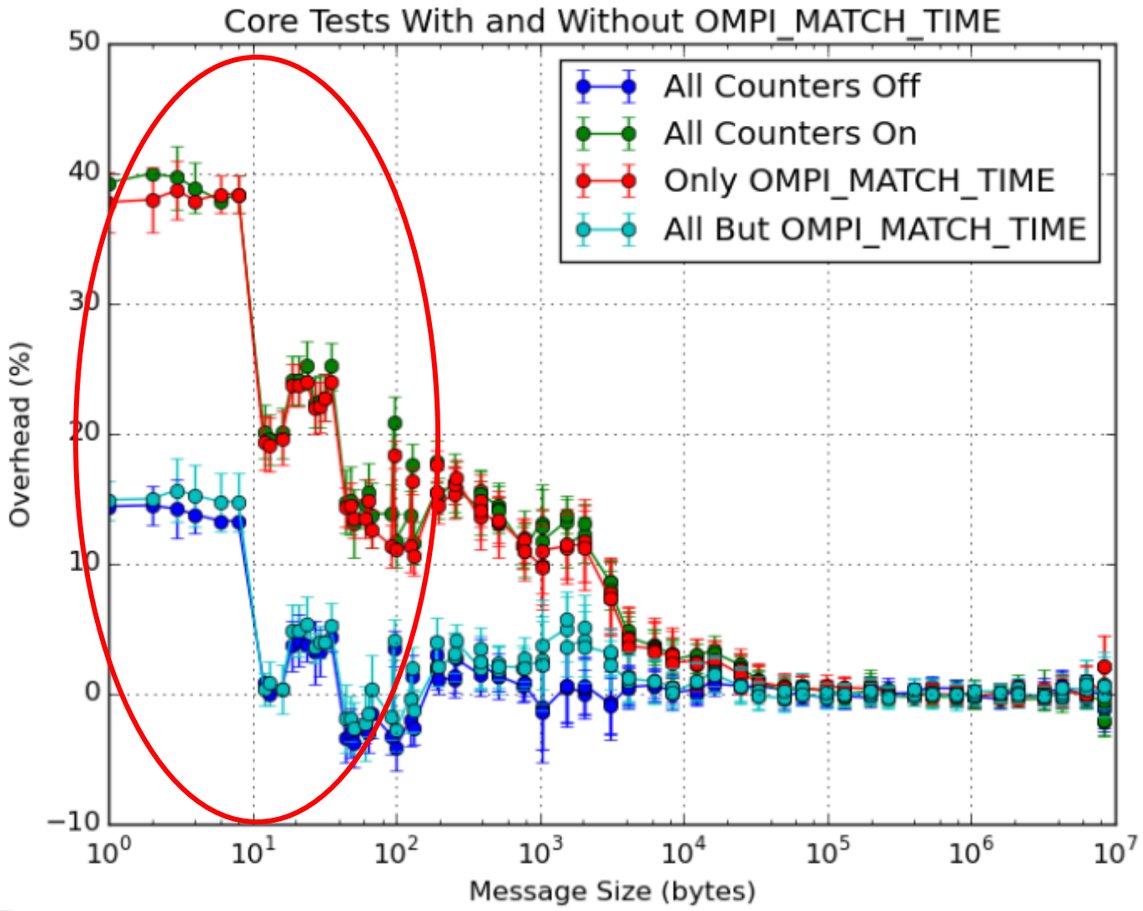
Oversubscription

- Highly oversubscribed environments
  - Support integration with any user-level threads packages



Performance events

- Expose information not available through other means
  - Out-of-sequence messages, time to match, number of unexpected
- Exposed as PAPI Software-Based Performance Counters
  - Easy integration with existing tools: TAU, Scalasca, …
  - Work to integrate them as MPI_T underway



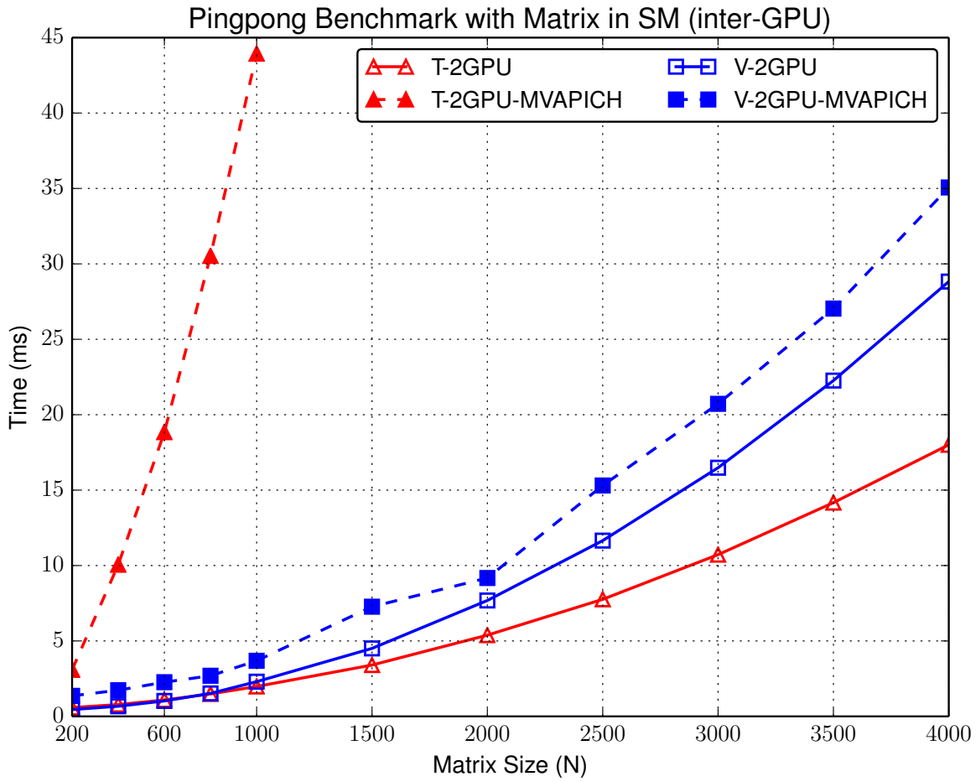
CUDA support

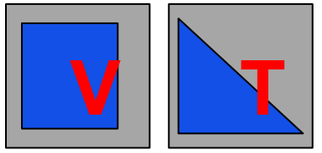
Point-to-point communications

- Multi-level coordination protocol based on the location of the source and destination memory
  - Support for GPUDirect
- Delocalize part of the datatype engine into the GPU
  - Provide a different datatype representation (avoid branching in the code)
  - Driven by the CPU
- Deeply integrated support for OpenIB and shared memory
  - BTL independent support available for everything else

Collective Communications

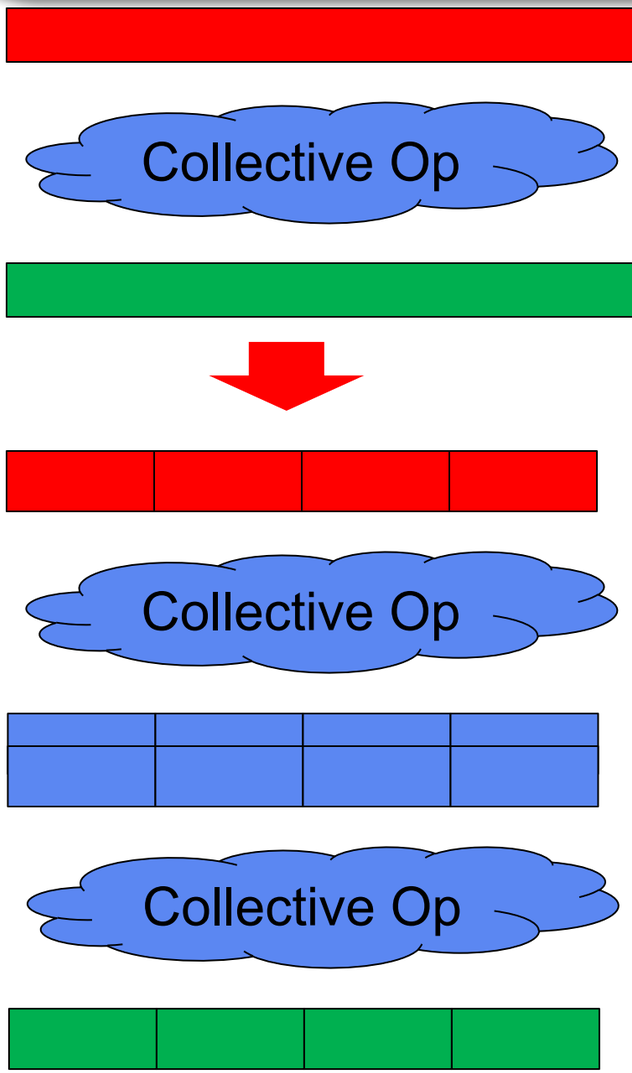
- Add support for collective operations, allowing to execute the collective communications directly on the GPU





Ivy Bridge E5-2690 v2 @ 3.00GHz, 2 sockets 10-core, 4 K40/node
MVAPICH 2.2-GDR

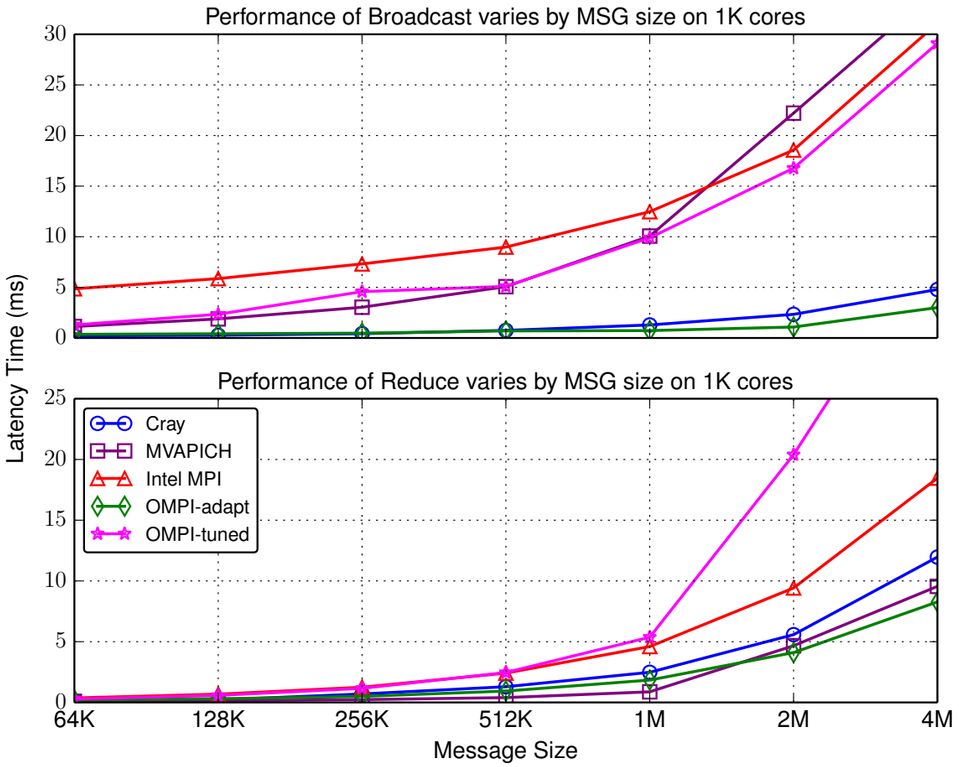
Collective communications

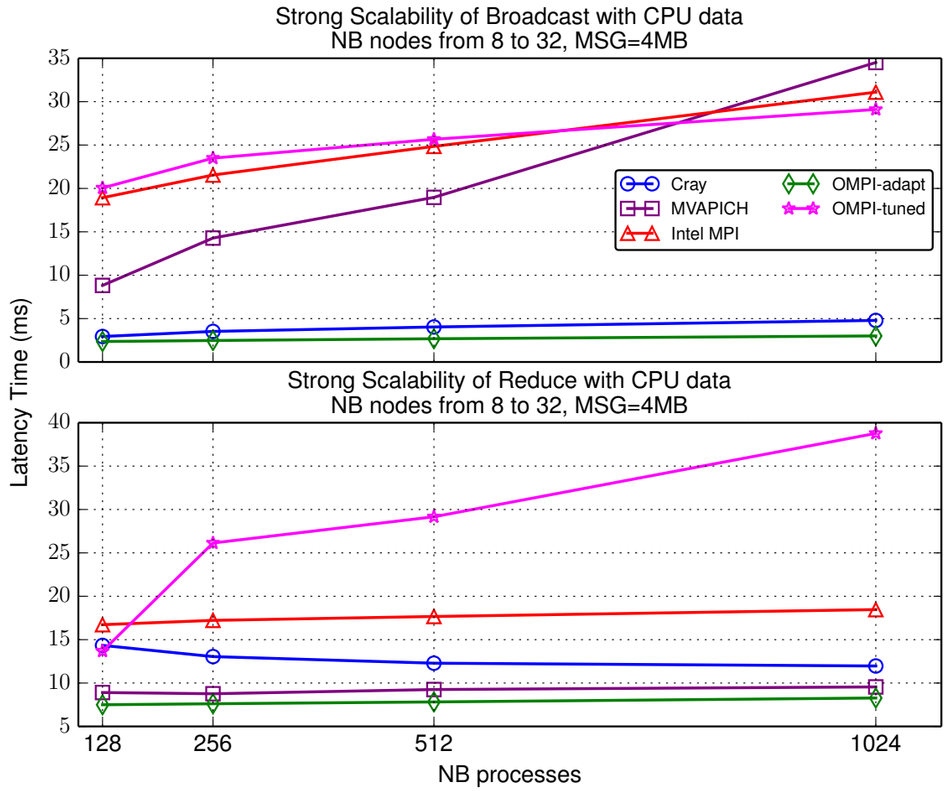


- Dataflow collective: different algorithms compose naturally (using a dynamic granularity for the pipelining fragments)
- Architecture aware: Each level reshape tuned collective to account for architecture capabilities
- The algorithm automatically adapts to network conditions
- Resistant to system noise

Collective: performance

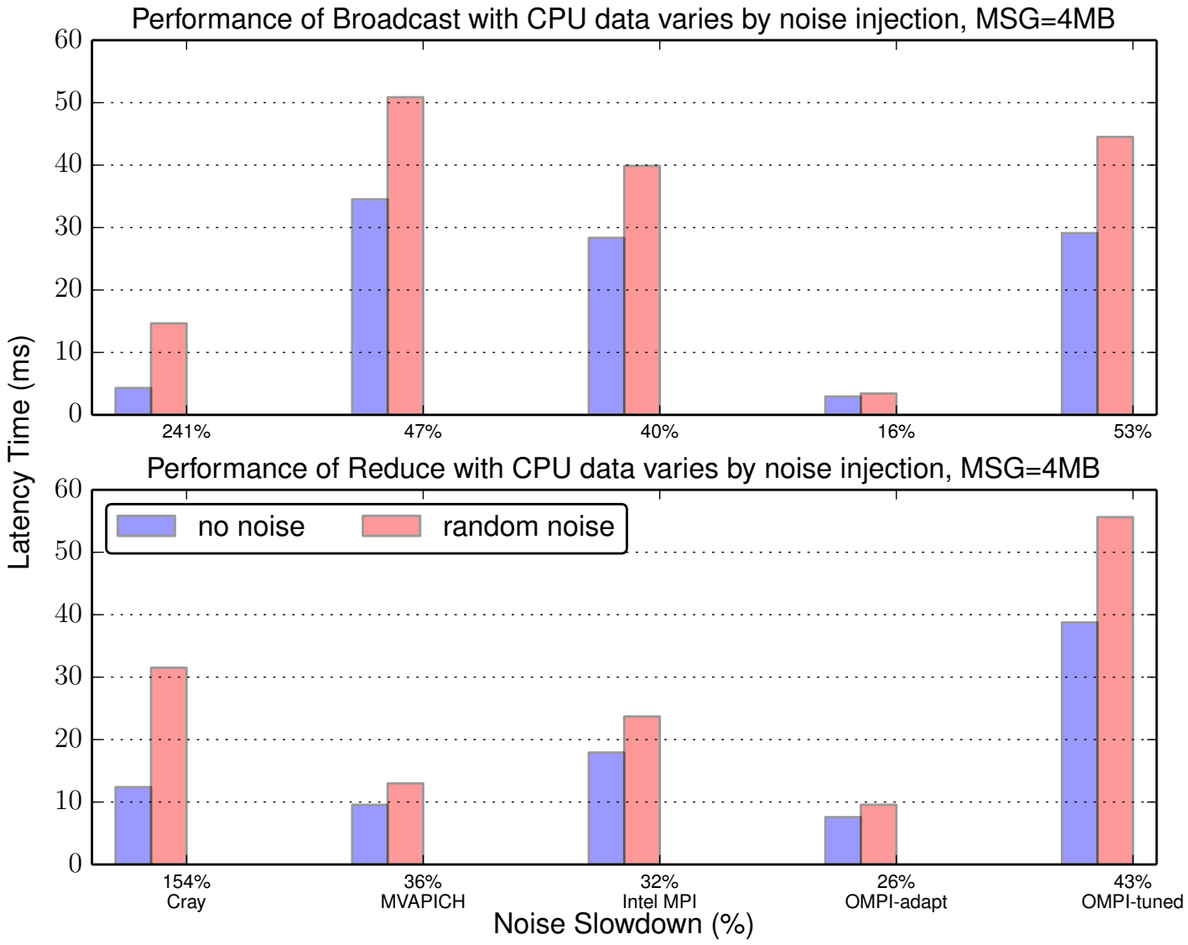
- NERSC's Cori (Cray XC40)

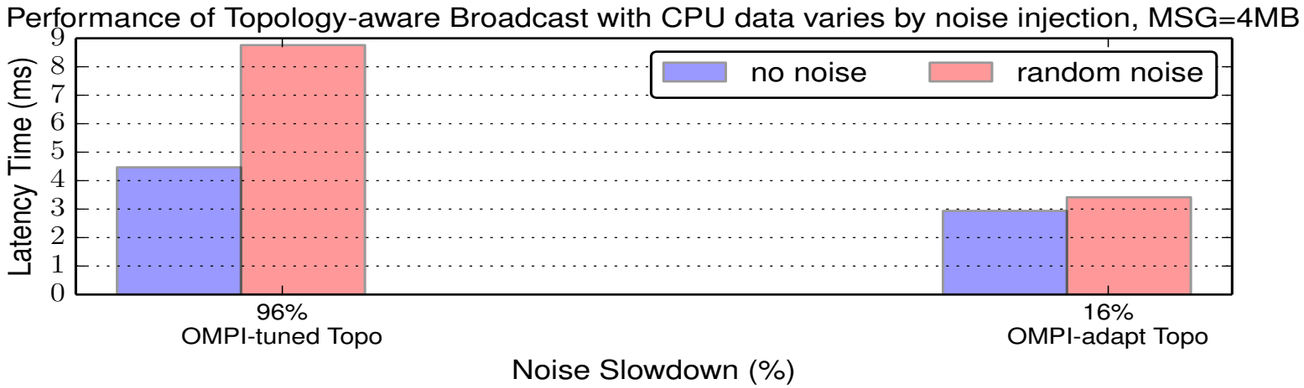




Collective: Noise Resistance

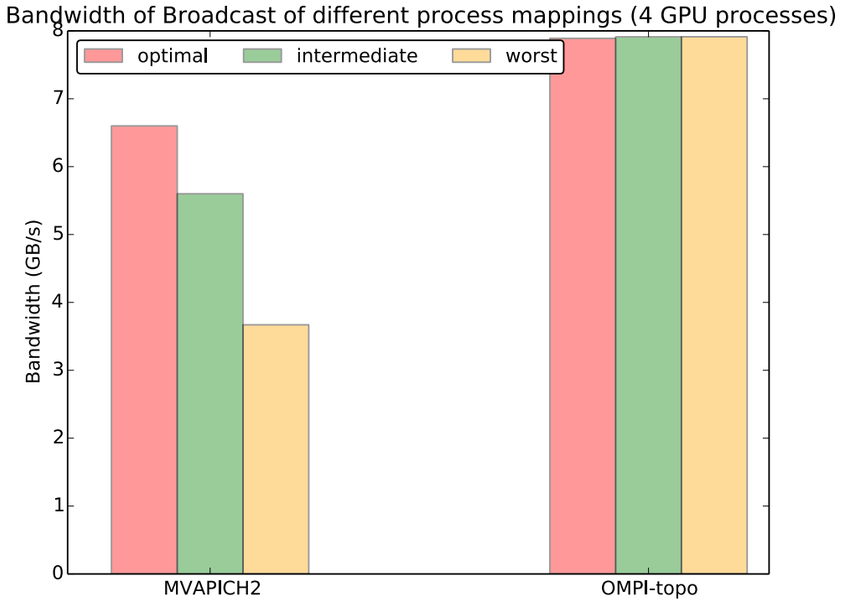
- Add noise to the application to see the impact on the collective performance



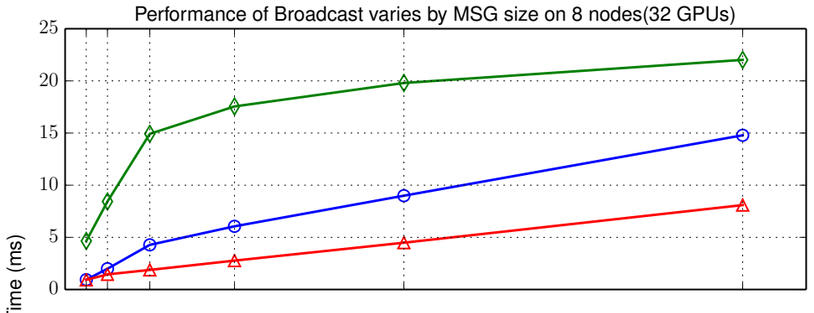


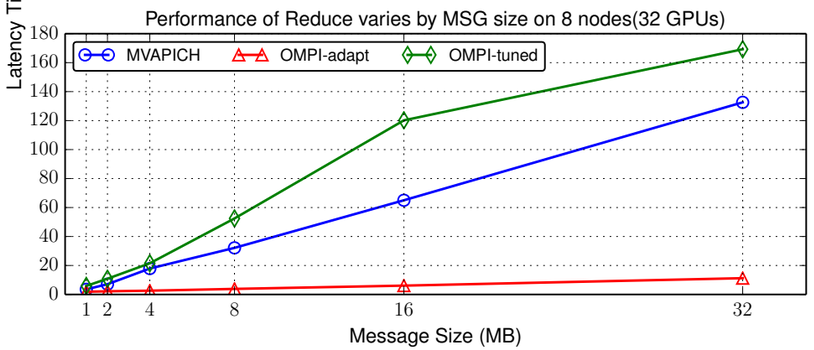
Collective: Hybrid Architectures

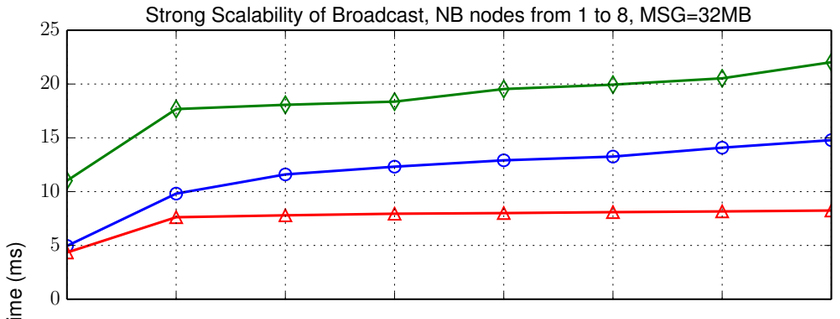
- Nvidia PSG machine

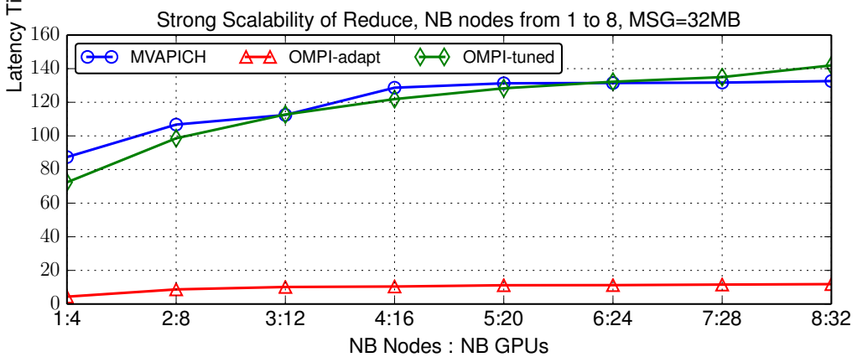


Collective performance independent of the process location





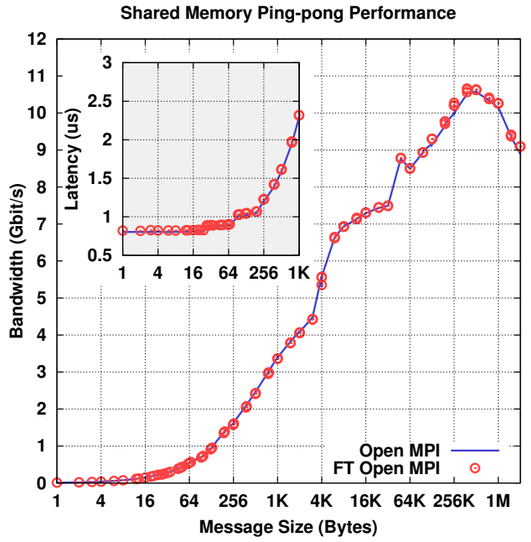




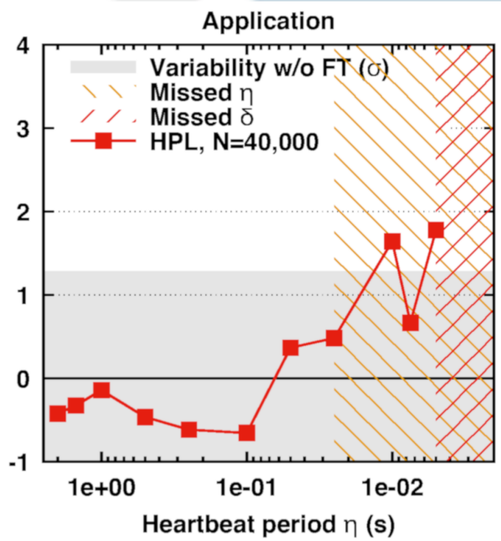
User level failure mitigation

- ULFM 2.0 released 11/03/17
  - Based on OMPI master (will remain in sync)
  - Transition to include it in master
- Scalable fault tolerant algorithms demonstrated in practice for revoke, agreement, and failure detection (SC'14, EuroMPI'15, SC'15, SC'16)

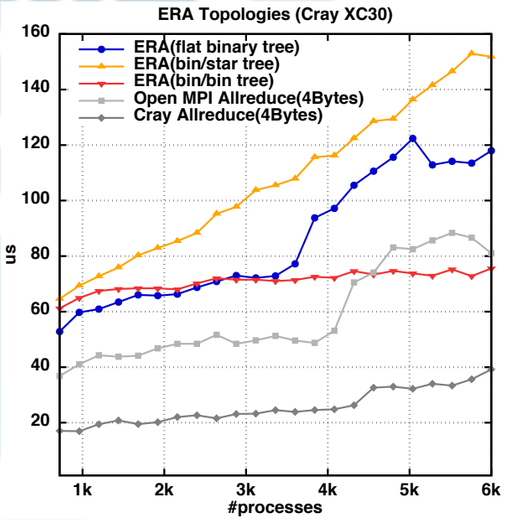
- Next steps:
- Make the underlying mechanisms available outside ULFM/OMPI
  - Move the failure detector and reliable broadcast in PMIx
  - SCONS a reliable communication infrastructure for PMIx



Point to point performance unchanged With FT enabled



Failure detector
(under 1/10 sec heartbeat)



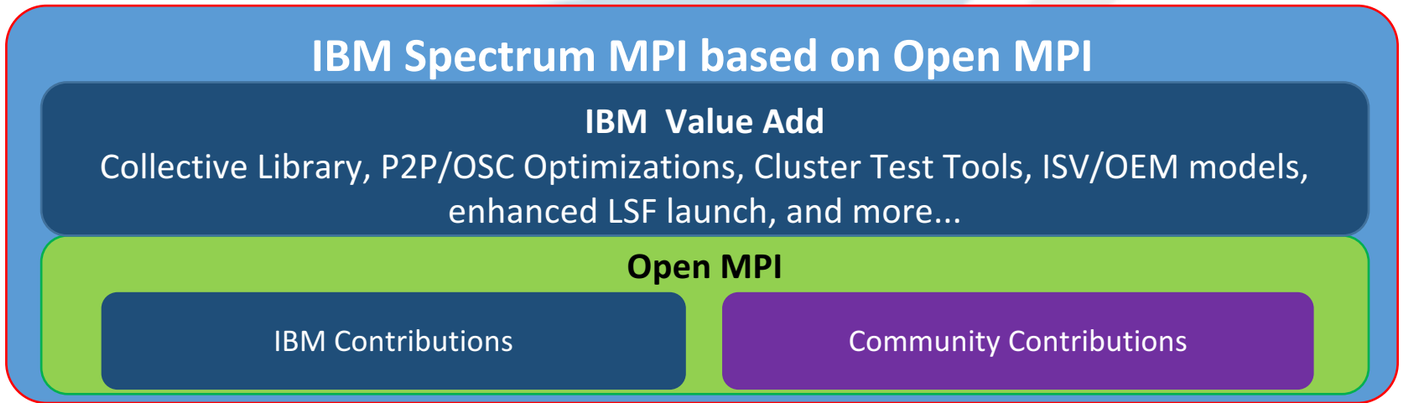
Fault Tolerant Agreement costs approximately 2x Allreduce

IBM Spectrum MPI

Josh Hursey

IBM Spectrum MPI

- IBM Spectrum MPI is a pre-built, pre-packaged version of Open MPI plus IBM value-add components.
  - Supports both PowerPC and x86 architectures
  - Supports most of Open MPI's components
- Spectrum MPI is based on Open MPI release branches
  - SMPI 10.1.0 based on OMPI v2.0.x branch
  - SMPI 10.1.1 based on OMPI v2.x branch (x86 only)
  - Upcoming SMPI 10.2.0 based on OMPI v3.0.x branch



Usability features

```
$$ mpirun -np 4 -host node01:2,node02:2 -prot –TCP ./hello
Host 0 [node01] ranks 0 - 1
Host 1 [node02] ranks 2 - 3

 host | 0    1
======|===========
    0 : shm  tcp
    1 : tcp  shm

Connection summary:
  on-host:  all connections are shm
  off-host: all connections are tcp

  0/  4) [node01] 61808 Hello, world!
  1/  4) [node01] 61809 Hello, world!
  2/  4) [node02] 100697 Hello, world!
  3/  4) [node02] 100698 Hello, world!
```

```
  [Affinity options]
    -aff on              : turn on affinity (bandwidth)
    -aff off             : turn off affinity (unbind)
    -aff v / -aff vv     : verbose
    -aff bandwidth       : interleave sockets
    -aff latency         : pack ranks
    -aff cycle:<unit>    : interleave binding over <unit>
    -aff width:<unit>    : bind each rank to an element of this
                           size <unit> can be hwthread, core,
                           socket, or numa.
    -aff default         : same as "bandwidth" above
    -aff auto[matic]     : same as "bandwidth" above
    -aff none            : same as "off" above
    -aff <option>,<option>,..  : comma separated list of above
```

```
[Interconnect selection]
  -PAMI  / -pami  : IBM PAMI
  -MXM   / -mxm   : Mellanox MXM
  -TCP   / -tcp   : TCP/IP
  -IBV   / -ibv   : OpenFabrics OFI
  -PSM   / -psm   : (x86) Intel PSM
  -PSM2  / -PSM2  : (x86) Intel Omni-Path
  -USNIC / -usnic : (x86) Cisco usNIC

[Additional PAMI options]
  -verbsbypass <ver>
  -pami_noib

[GPU support]
  -gpu            : Enable GPU awareness in PAMI

[On-host communication method]
  -intra nic   : use off-host BTL for on-host traffic
  -intra vader : Open MPI's vader shared memory BTL
  -intra shm   : alias for -intra vader

[IP network selection]
  -netaddr <spec>,<spec>,..
  -netaddr <type>:<spec>,<spec>,..
  <type> can be any of
      rank       :  MPI traffic
      control    :  Control traffic (out-of-band)
      mpirun     :  synonym for "control"
  <spec> can be either
      interface name  :  eg eth0 or ib0 etc
      CIDR notation   :  eg 10.10.1.0/24
```

```
$$ mpicc --version
IBM XL C/C++ for Linux, V13.1.5 (5725
Version: 13.01.0005.0000
$$
$$ mpixlc --version
IBM XL C/C++ for Linux, V13.1.5 (5725
Version: 13.01.0005.0000
$$
$$ mpipgicc --version

pgcc 16.10-0 linuxpower target on Lin
The Portland Group - PGI Compilers an
Copyright (c) 2016, NVIDIA CORPORATIO
reserved.
$$
$$ OMPI_CC=gcc mpicc --version
gcc (GCC) 4.8.5 20150623 (Red Hat 4.8
Copyright (C) 2015 Free Software Foun
```

Value-add features

- `MPI_ROOT`
  - Can have multiple installed versions; switch between them at runtime by setting a single environment variable
- Compiler support (single install)
  - GNU, XL (power), PGI (power), Intel (x86)
- Multiple concurrent PMPI interface wrapping
  - **`-entry/-entrybase`** options
- PAMI over IB verbs
  - Verbs by-pass feature further improves latency for small messages
- **`libcollectives`** library with advanced selection logic
  - 1.6x-11.8x faster than Open MPI's 'basic' and 'tuned' over PAMI
- Lightweight core files

IBM Testing and Support

- Extensive level of testing for IBM releases
  - Standard Open MPI release testing…
  - …Plus Platform MPI test suites
  - …Plus HPC stack integration testing
- IBM Customer Support
  - For customers running a licensed copy of IBM Spectrum MPI
  - IBM will work with customers and partners to resolve issues in non IBM-owned components
- Contribute to community testing
  - MTT nightly regression testing on IBM PowerPC servers
  - Jenkins CI testing on IBM PowerPC servers

CORAL – Summit & Sierra

- Delivering **more than 100 petaFLOP/s** peak performance by combining **IBM POWER9 CPUs + NVIDIA Volta GPUs + Mellanox EDR InfiniBand**
- Spectrum MPI will play a critical role in application performance at scale on ORNL's Summit and LLNL's Sierra systems
  - Lots of work on MPI point-to-point, collective, and one-sided performance and resource consumption as applications scale.
- The Job Step Manager (JSM) replaces ORTE as the job launcher.
  - PMIx 2.x compliant runtime project tuned for the IBM LSF/CSM computing environment.
  - Focus on fast job launch, and managing multiple concurrent jobs within a single LSF allocation.

Charting the PMIx Roadmap BoF
Thursday, Nov. 16
12:15-1:15pm

CORAL COLLABORATION
OAK RIDGE • ARGONNE • LIVERMORE

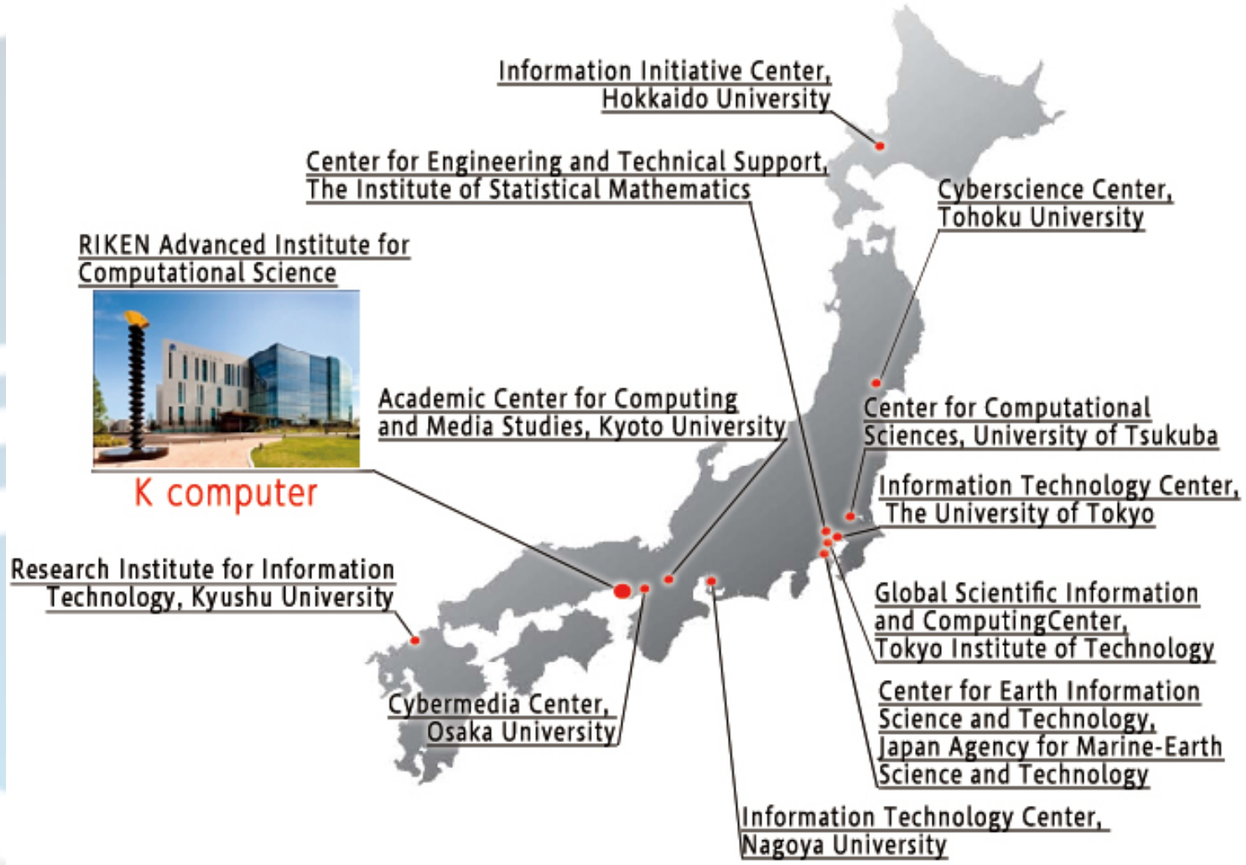
IBM Spectrum MPI

OMPI BOF – a user perspective

Gilles Gouaillardet <gilles@rist.or.jp>

Open MPI @ HPCI

- High Performance Computing Infrastructure (HPCI)
  - Connects flagship K computer and other major supercomputers in Japan
  - XSEDE-like (or PRACE-like) in Japan
- Open MPI is indirectly used on Fujitsu systems (Linux / Sparc / ToFu) via Fujitsu MPI.
- HPCI is available free of charge to worldwide researchers
- Visit us at booth #219!



Open MPI information

- Official website
  - § https://www.open-mpi.org
- Three mailing lists
  - § announce@lists.open-mpi.org
  - § users@lists.open-mpi.org
  - § devel@lists.open-mpi.org
- Github repository
  - § https://github.com/open-mpi/ompi

Open MPI support

- Community based support for vanilla Open MPI
- Contact vendors (Bull, Fujitsu, IBM, Cisco, Mellanox, etc.) directly if you are using an Open MPI based vendor MPI

users@lists.open-mpi.org 1/2

- The best way to contact us !
  - All developers read and answer questions
  - End users often share their experience
  - Questions are generally replied the same day
  - All questions are treated equally
- The right place to
  - Ask about an Open MPI feature
  - Report a bug
  - Request a feature

users@lists.open-mpi.org 2/2

- This list is for Open MPI only
  - MPI standard should be discussed at the MPI forum http://mpi-forum.org
  - General MPI questions (non specific to Open MPI) can be discussed elsewhere (http://stackoverflow.com for example)
  - Bug reports for vendor MPI based on Open MPI
  - Bug reports for other MPI libraries

Frequently Answered Questions

- Some questions are frequently asked by the community
- Here is a short overview with answers

MPI task binding

- Still confusion about what the (default) process-to-core binding is about
- 3 concepts in Open MPI
  1. Mapping: assign each task to a location
  2. Binding: bind task to specific processing element
  3. Ranking: assign MPI_COMM_WORLD rank to each task
- Refer to Ralph Castain's SC'16 slides https://www.open-mpi.org/papers/sc-2016/
  - Starting with slide #84

One-sided communications

- RMA semantic is complex and raised a lot of questions (bugs vs. user mistakes)
- Semantic might not be what you expect, nor what you want
- Do not hesitate to
  - (Re)Read the MPI standard
  - Try another MPI library and check the behavior
  - Do not jump too quickly to any conclusion

Open MPI and PBS Pro / Torque

- Open MPI must be built with `tm` support
- By default, Open MPI *tries* to build `tm` support
  - Use `--with-tm` to have configure fail if `tm` is not available
- Otherwise a multi node job will end up using all the resources of a single node
  - …and may even fail to start because not enough slots are available

New default IO component

- ROMIO (from MPICH) used to be the default MPI IO component
- OMPIO (specific to Open MPI) is the new default, unless a Lustre filesystem is used
- `ompio` is a less mature than the well established ROMIO
  - You can use the `--mca io romio314` mpirun parameter to force using the ROMIO component

PGI and SLURM

- SLURM is likely built with gcc
- `libpmi.la` likely sets the `-pthread` flag
- `pgcc` cannot build Open MPI because `-pthread` is not a valid option

```
$ cat  /usr/pppl/pgi/17.3/linux86-64/17.3/bin/siterc
#####################################
# siterc for gcc commands PGI does not support
#####################################
switch -pthread is
        append(LDLIB1=-lpthread);
```

Docker and Open MPI

- Docker is great at containers, but was not designed with HPC in mind
  - Open MPI is container agnostic
  - Can run Open MPI inside Docker containers (with no special configuration)
  - Open MPI has no specific support for Docker
- http://singularity.lbl.gov was designed with HPC in mind and is fully supported by Open MPI

Open MPI for Exascale (OMPI-X)

David E. Bernholdt, ORNL

for the OMPI-X team

OAK RIDGE National Laboratory

THE UNIVERSITY OF TENNESSEE KNOXVILLE

Sandia National Laboratories

Los Alamos NATIONAL LABORATORY EST. 1943

Lawrence Livermore National Laboratory

ECP EXASCALE COMPUTING PROJECT

Acknowledgements

- **This research was supported by the Exascale Computing Project (17-SC-20-SC), a joint project of the U.S. Department of Energy's Office of Science and National Nuclear Security Administration, responsible for delivering a capable exascale ecosystem, including software, applications, and hardware technology, to support the nation's exascale computing imperative.**
- This work was carried out in part at Oak Ridge National Laboratory, managed by UT-Battelle, LLC, for the U.S. Department of Energy under contract number DE-AC05-00OR22725.
- Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc. for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.
- This work was performed in part at Los Alamos National Laboratory, supported by the U.S. Department of Energy contract DE-FC02-06ER25750.
- Part of this work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

ECP: Exascale Computing Project

- From https://exascaleproject.org (emphasis mine)…

  ECP is chartered with accelerating delivery of a capable exascale computing ecosystem *to provide breakthrough modeling and simulation solutions to address the most critical challenges in scientific discovery, energy assurance, economic competitiveness, and national security.*

  This role goes far beyond the limited scope of a physical computing system. ECP's work encompasses the development of an entire exascale ecosystem: applications, system software, hardware technologies and architectures, along with critical workforce development.

- Funded by DOE Office of Science and NNSA, managed by the DOE laboratories
  - With participation by other government agencies
- "Exascale" defined as 50x performance of current systems on applications
- Expecting initial exascale system delivery in 2021

Our Project: Open MPI for Exascale (OMPI-X)

- A project within the ECP Software Technologies (ST) / Programming Models and Runtimes (PM) area
- Ensure that the MPI standard and its specific implementation in Open MPI meet the needs of the ECP community in terms of performance, scalability, and capabilities or features
  - Participating in the MPI Forum to address the needs of ECP applications and libraries
  - Working within the Open MPI community to
    - Prototype and demonstrate exascale-relevant proposals under consideration by the MPI Forum
    - Improve the fundamental performance, scalability, and architectural awareness of Open MPI, particularly for exascale-relevant platforms and job sizes
- *The ECP "Exascale MPI" project focuses on MPICH*

The OMPI-X Team

- ORNL
  - **David Bernholdt** (Lead PI)
  - Manju Gorentla Venkata
  - Terry R. Jones
  - Thomas J. Naughton III
  - Geoffroy R. Vallee
- LANL
  - Nathan Graham
  - Evan Harvey
  - Nathan Hjelm
  - **Howard Pritchard**
- LLNL
  - Chris Chambreau
  - Murali Emani
  - Ignacio Laguna
  - **Martin Schulz**
- SNL
  - **Ron Brightwell**
  - Ryan Grant
- UTK
  - **George Bosilca**
  - Aurelian Bouteiller

OMPI-X Focus Areas

- **Runtime Interoperability for MPI+X and Beyond** [Vallee]
  - APIs for better sharing of threads between MPI and other thread-based runtimes
  - Intend collaboration with ExaMPI [MPICH] and SOLLVE [OpenMP]
- **Extending the MPI Standard to Better Support Exascale Architectures** [Grant]
  - Endpoints, Finepoints, Sessions
- **Open MPI Scalability and Performance** [Gorentla]
  - Memory footprint, collectives, message matching, one-sided, PMIx
  - See also George's UTK presentation
- **Supporting More Dynamic Execution Environments** [Jones]
  - Intelligent process placement and contention management
- **Resilience in MPI and Open MPI** [Bosilca]
  - ULFM, ReInit, resilience in PMIx
  - **See George's UTK presentation**
- **MPI Tool Interfaces** [Schulz]
  - MPI_T, PMPI replacement
- **Quality Assurance for Open MPI and New Developments** [Pritchard]
  - Test infrastructure deployed to ECP-relevant systems
  - Regular testing of Open MPI and OMPI-X developments

Survey Says…

- The OMPI-X project recently conducted a survey of the ECP Application Development (AD) and Software Technology (ST) projects
- Survey questions covered a range of topics: Application demographics, non-MPI applications, basic performance characterization, MPI usage patterns, MPI tools ecosystem, memory hierarchy details, accelerator details, resilience, use of other programming models, MPI with threads
- Received a total of 77 responses (project level), 56 of which use MPI
- Talk presented at ExaMPI Workshop paper on Sunday, paper to appear in special issue of Concurrency and Computing: Practice and Experience
  - A Survey of MPI Usage in the U.S. Exascale Computing Project, David E. Bernholdt (ORNL), Swen Boehm (ORNL), George Bosilca (UTK), Manjunath Gorentla Venkata (ORNL), Ryan E. Grant (SNL), Thomas Naughton (ORNL), Howard P. Pritchard (LANL), Martin Schulz (LNLL, TU Munich), Geoffroy R. Vallee (ORNL)
- Considering broadening survey and opening it to the wider community

Runtime Interoperability for MPI+X and Beyond

Motivation and Goals

- Both MPI and OpenMP runtimes use threads, but there is no coordination
- Optimal placement of MPI ranks and threads is therefore difficult on complex architectures
- Investigate runtime coordination for optimal placement of threads and MPI ranks
  - Data exchange between runtimes
  - Implement optimized placement policies
- Eventually, generalize to other node-level threading models

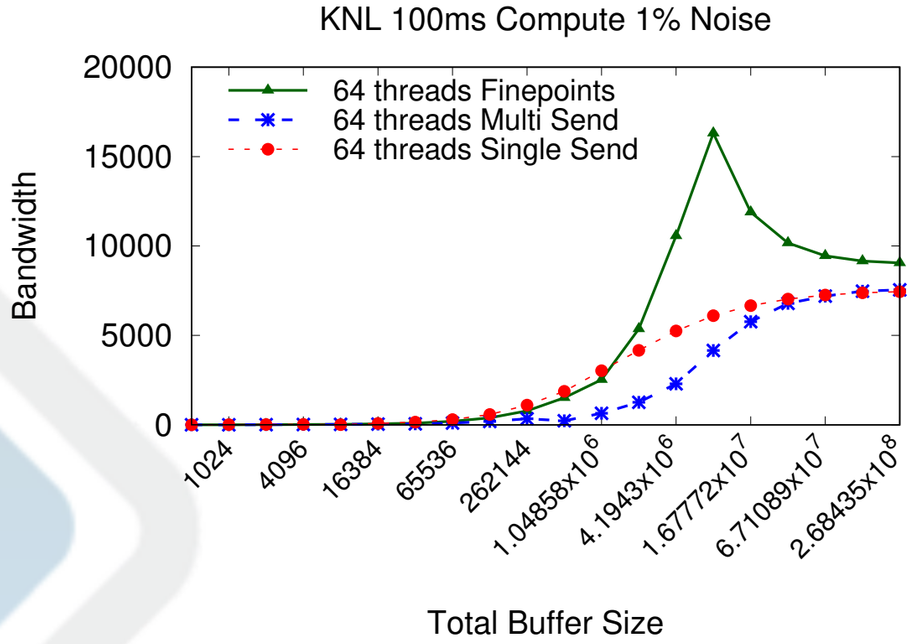
Recent Progress

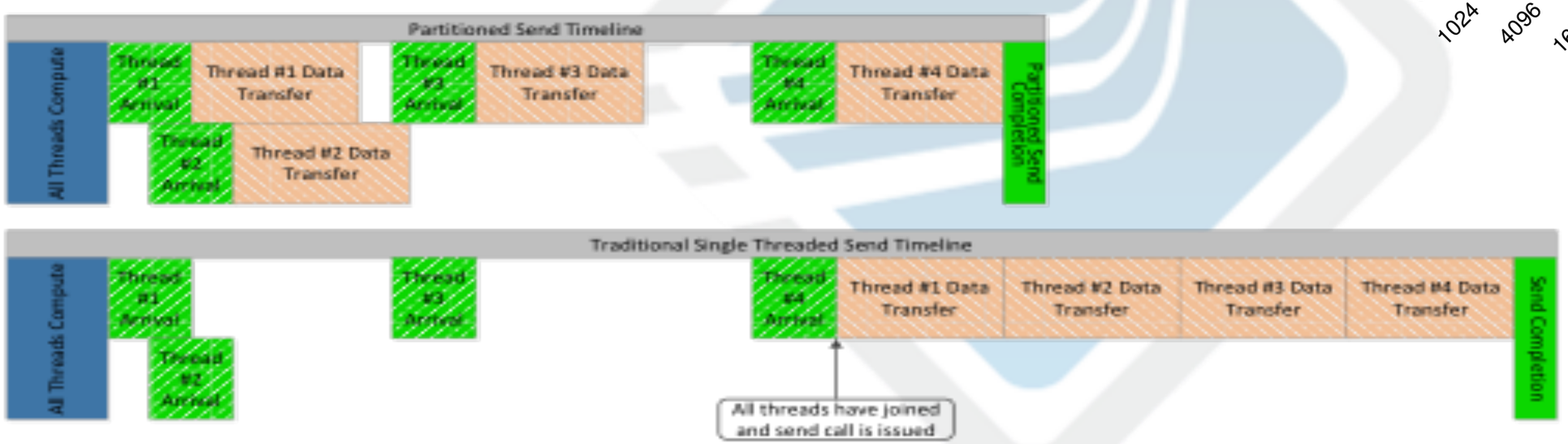
- Modify the OpenMP LLVM compiler to interface with PMIx (Open MPI and some resource managers already rely on PMIx)
- Data exchange between the MPI and OpenMP runtimes via PMIx
- Implement a placement policy based on the number of MPI ranks and available cores/HT per node
- Upcoming work: evaluation and implementation of more advanced policies (collaboration with ECP SOLLVE project)

Survey:
- 86% of projects plan to use multiple threads per rank
- 45% use OpenMP; 21% use Kokkos or RAJA

Finepoints

- Finepoints provides efficient multi-threading for MPI
- Each thread sends a portion of a message
- MPI aggregates partitions and send messages efficiently
- Allows for early-bird overlapping where data can be sent before a traditional fork-join-send model
- Initial results on KNL promising, allowing high ”perceived bandwidth”





Early-bird Communication Example

Survey: 52% of projects do not need thread-level addressability on the target

Implement, demonstrate, and evaluate prototype of MPI Sessions proposal

Goals

- The proposed Sessions extensions to the MPI standard is intended to provide a tighter integration with the underlying runtime used by an MPI implementation, as well as provide a more scalable mechanism for applications to specify communication requirements than is currently supported by the MPI standard.

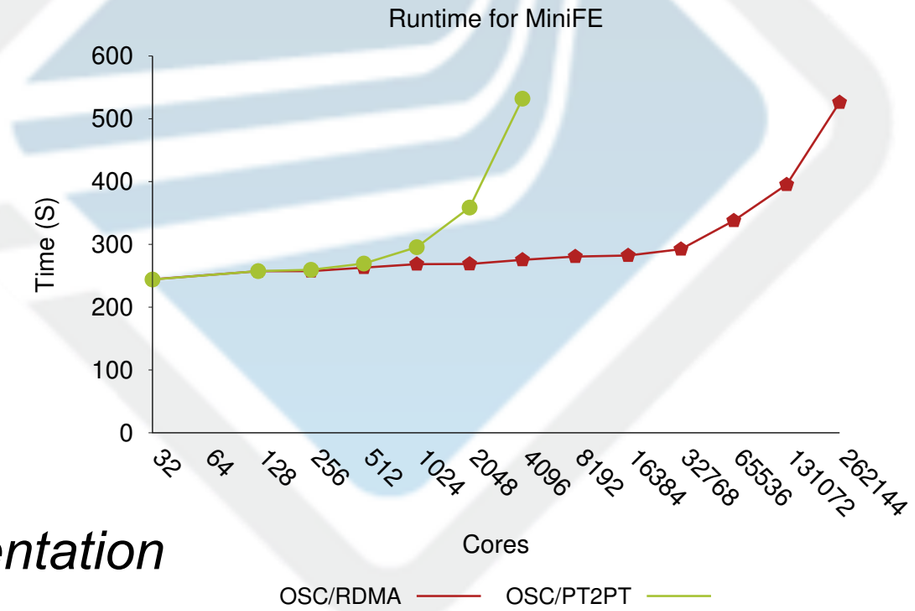
Survey: interest in job-to-job communication capabilities, could be facilitated by Sessions

Recent Progress

- The Sessions working group has been using feedback from Martin Schulz's presentation at the September '17 MPI Forum to consider alternatives to the original Sessions proposal.
  - The WG is considering reusing concepts from the endpoint proposal to support important Sessions concepts like isolation, etc.
- The WG is working with the PMIx group to ensure PMIx will have hooks in place to support implementing Sessions (or whatever it ends up being called) in Open MPI - https://github.com/pmix/pmix/pull/69

OpenMPI Performance Improvements

- Developed native one-sided (RMA) component for OMPI
- Significantly improved performance over previous method
- MiniFE now scales to full size of Trinity Haswell nodes
- Latencies and throughputs are comparable to vendor optimized MPI



See also George Bosilca's presentation

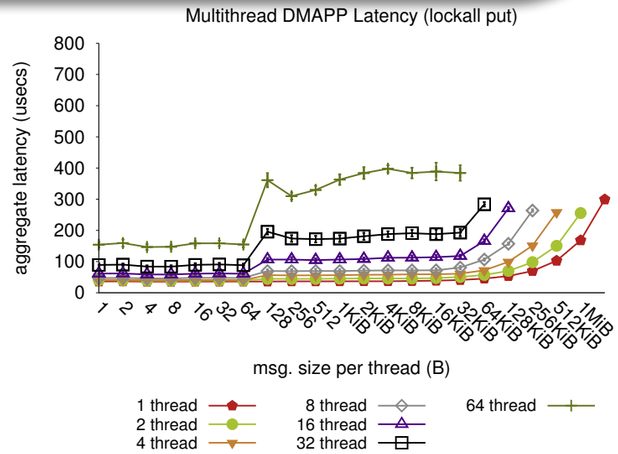


Fig. 1: DMAPP latency put-lockall

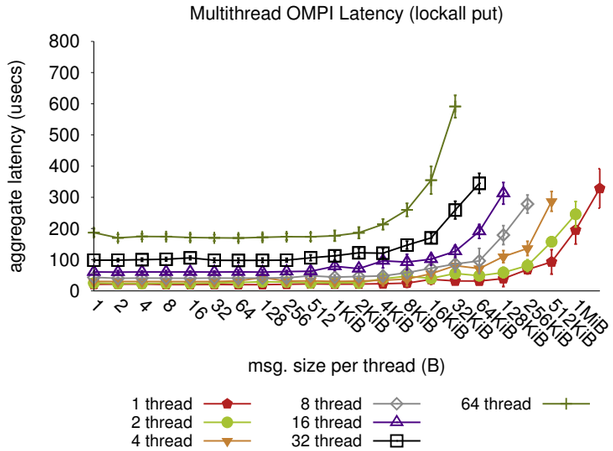
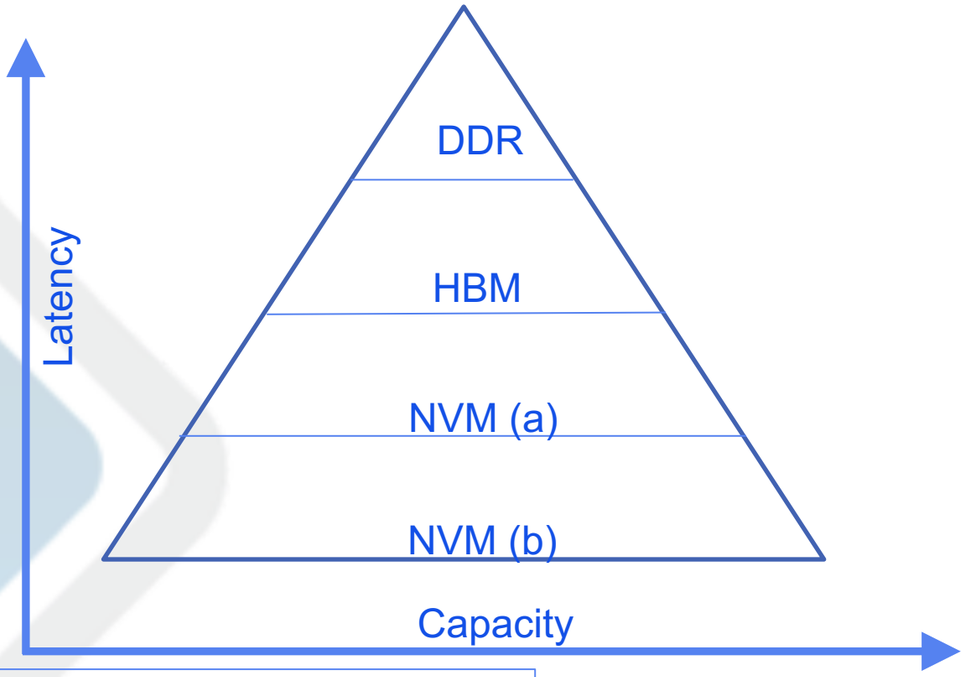


Fig. 2: OMPI latency put-lockall

Complex Memory Hierarchies

Motivation and Goals

- Architectural Awareness
- Adapting to Fabric including Topology and Concerns
- Adapting to Deep Memories and new Memory Layers
- Take advantage of available architecture strengths & do it automatically when possible



Survey:

- 73% of projects expect to explicitly manage memory placement and movement
- 46% of projects expect to move data between different memory spaces between local and remote nodes
  - For example, main memory on source to non-volatile memory on destination

MPI Tools Interfaces

- Replacement of the PMPI interface
  - Application developers see a need for multiple tools at run time
  - MPI Forum Tools Working Group is discussing how to provide the ability to intercept MPI calls and pass execution to multiple tools
  - OMPI-X Milestone addresses API development and prototyping
- MPI_T performance variable and event interface
  - There is interest in internal MPI profiling data
  - Software-based counters are being added as MPI_T performance variables (as described by Eberius, Patinyasakdikul, Bosilca)
  - OMPI-X Milestone expands available performance variables and prototypes MPI_T event interface as per the Tools Working Group

Survey:
- 27% of projects need to be able to use multiple “tools” simultaneously
- 52% of projects “interested” or “very interested” in MPI_T data
  - Load balance, memory use, and message queue info
  - Function call time, network counters

Continuous Integration Testing Infrastructure for Open MPI

Goals

- Enhance Continuous Integration and Nightly testing where required to ensure OMPI-X contributions to Open MPI are being sufficiently validated for correctness and performance
  - Especially on DOE exascale early access systems
- Ensure Open MPI works well with ECP's Spack-based install mechanism

Recent Progress

- Helped resolve problems with using the Python client with the Open MPI MTT database server
- Investigating use of the Java Web Start approach for connecting a slave node to a Jenkins server in cases where the user neither has root privilege (no access to systemd), and where the front end nodes do not allow for persistent crontab entries.
  - See https://github.com/open-mpi/ompi/wiki/Jenkins-Build-Agent
- For greater flexibility for Spack based installs, added an independent PMIx Spack package.
  - Keeping Spack's Open MPI package up to date with Open MPI releases.

Short community feature updates

Quick updates on

Open MPI technology

PMIx in OMPI

PMIx BoF: Thursday, Nov 16, 12:15-1:15pm

Room 210-212

(yes, that's tomorrow!)

PMIx

- Process Management Interface – Exascale
  - Underlying run-time support for exascale applications
  - Client – server architecture
  - Standalone community, development
- Not just targeted at exascale

Embedded PMIx versions

Open MPI version	PMIx version
v2.0.x	v1.1.5
v2.1.x	v1.2.x
v3.0.x	v2.0.x
v3.1.x	v2.1.x
Git master	v3.0.x

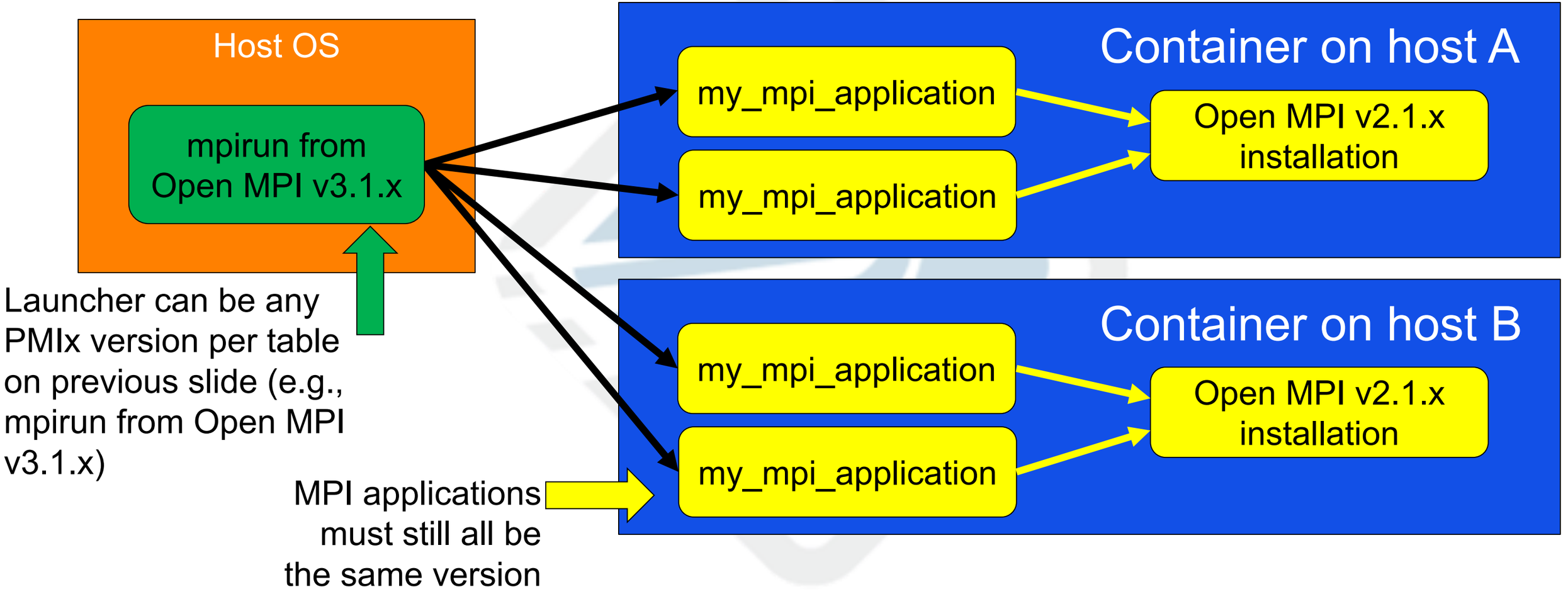
Cross-version mpirun interoperability

Open MPI version	PMIx version
v2.0.x	v1.1.5
v2.1.x	v1.2.5+
v3.0.x	v2.0.3+
v3.1.x	v2.1.x
Git master	v3.0.x

Server >= Client

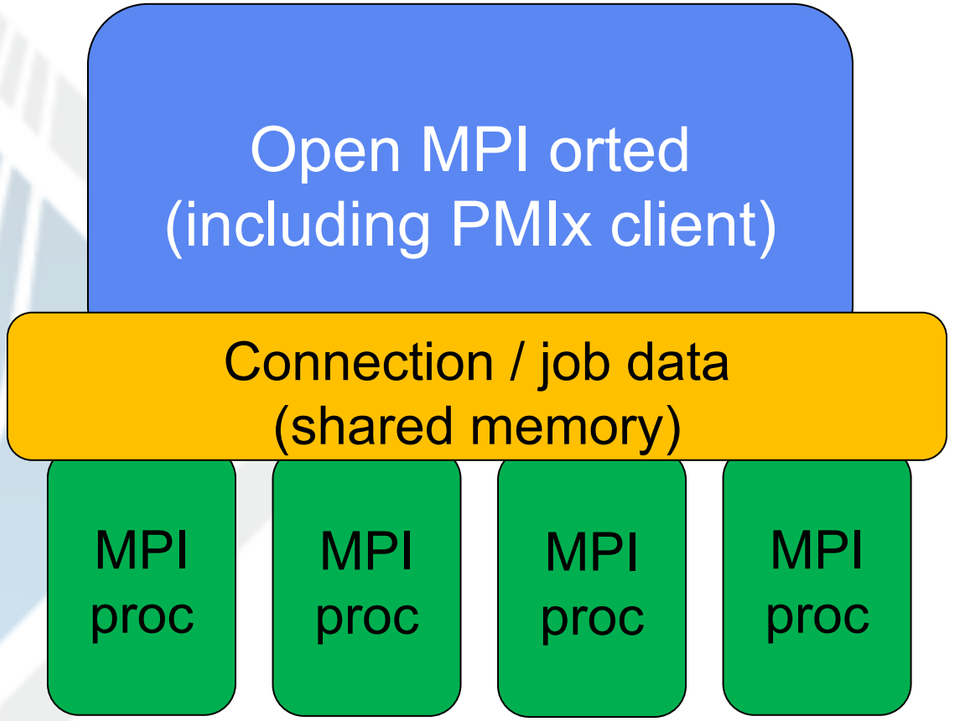
Any client/server combination

Cross-version mpirun interoperability



Scalable startup

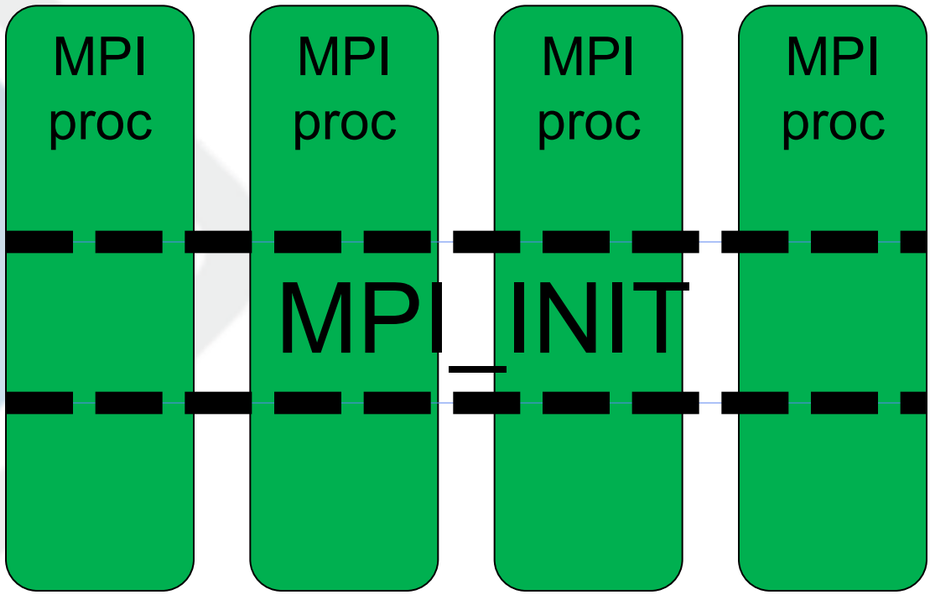
- Many-core node support (memory footprint)
  - Connection and job data is stored once per node
  - Shared memory access given to application procs



Scalable startup

- Many-node support (init time)
  - Eliminate OOB barriers during MPI_INIT
  - Connection info exchange:
    - **`pmix_base_async_modex=1`**
    - **`pmix_base_collect_data=0`**
  - Sync barrier at end of MPI_INIT:
    - **`async_mpi_init=1`**

Only suitable for sparsely connected apps

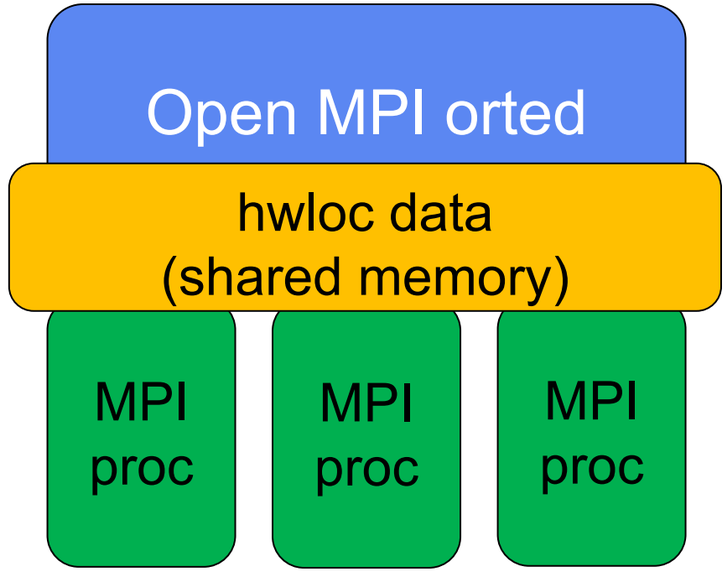


Hardware locality (“hwloc”)

Brice Goglin

Reduced memory footprint
(and launch time) on many-cores

- hwloc uses ~1MB per process on KNL
  - Bad when using 1 process per core
  - Even worse on upcoming many-core platforms
- Topology may now be in **shared-memory**
  - Only 1MB **per host**
  - 64-rank launch on KNL down from 4.2s to 1.9s
  - No need to exchange/parse XML anymore
- Available in upcoming hwloc 2.0
  - Supported in Open MPI 3.1



hwloc v2.0 coming soon

- Many changes to support heterogeneous and hybrid memories
  - KNL MCDRAM, NVDIMM, etc.
- API cleanup
  - **Not ABI compatible with hwloc v1.x!**
- Planned for Q1 2018
  - **Please try porting your code to v2.0 NOW**
  - Report issues ***before the v2.0 release***
  - Git master snapshots available online

Open MPI on ARM

arm

ARM64 support

- Mainly focused on ARMv8 (aarch64)
- Continues integration on ThunderX1 ARM with InfiniBand EDR at HPC Advisory council cluster center
- Tested with UCX framework
- CY18: MTT testing in collaboration with Los Alamos

Los Alamos

Los Alamos
NATIONAL LABORATORY
EST. 1943

SLURM-related changes

- DOE trilabs encountered problems with signal forwarding to jobs when using Open MPI 2.x and newer with SLURM 17.0.2
- *scancel* didn't work
  - MPI processes on head node of *mpirun* launched job didn't see signal, neither did *mpirun*

SLURM-related changes

- Problem had to do with SLURM not knowing anything about mpirun and its child processes on the head node (not fork/exec'd by a *slurmstepd*)
- Option added to change *mpirun* behavior to not launch local MPI processes directly, but go through SLURM using an MCA parameter:
  - **`ras_base_launch_orted_on_hn`**
  - Defaults to false on non-Cray XE/XC systems, true for Crays
- Note this option may add more *jitter* on head node

PSM2 MTL changes

- Added a set of MPI_T control variables to proxy for PSM2 environment variables
- Added a set of MPI_T performance variables allowing access to PSM2 runtime statistics (via *psm2_mq_get_stats*) using MPI_T

Open MPI and Fujitsu

Fujitsu Limited

FUJITSU

MPI for the Post-K computer

- Fujitsu is developing an MPI library for the post-K computer based on Open MPI
  - Currently based on OMPI 2.0.3
- Contribution to Open MPI from post-K MPI
  - Persistent collective communication request [*now working; waiting standardization in MPI Forum*]
  - Datatype for half-precision floating point [*in 2018*]
  - Improved Java binding [*done*]
  - And more [*Thread parallelization, Hang-up Detection, MPI-related Statistical Information for Application Tuning*]

Community support

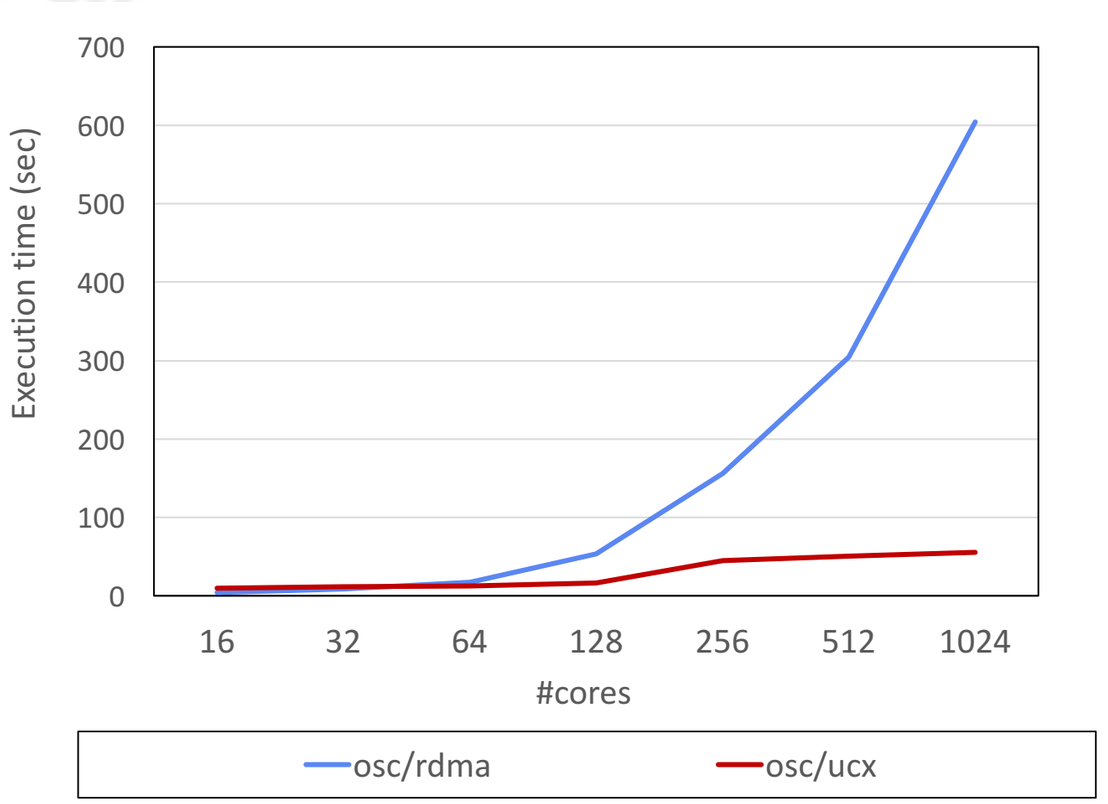
- Continue collaboration with Open MPI community
  - Reduced memory footprint (optimized key store, dynamic add_proc)
  - ARM / SPARC support
  - PMIx support (integration with 2.0)
- Continue quality activity
  - Bug fixes
  - Testing
    - Well-tested Open MPI for ARM [*now working; expected in 2018*]

Mellanox Technologies

Mellanox's HPC-X based on Open MPI

- Add InfiniBand accelerated libraries using latest capabilities
  - Point-to-point acceleration: UCX
  - Collective communication Acceleration: HCOLL
- Recent MPI features accelerated
  - RMA
  - Atomics
  - Tag matching: HW support
  - Collective operations: Allreduce, Allgatherv

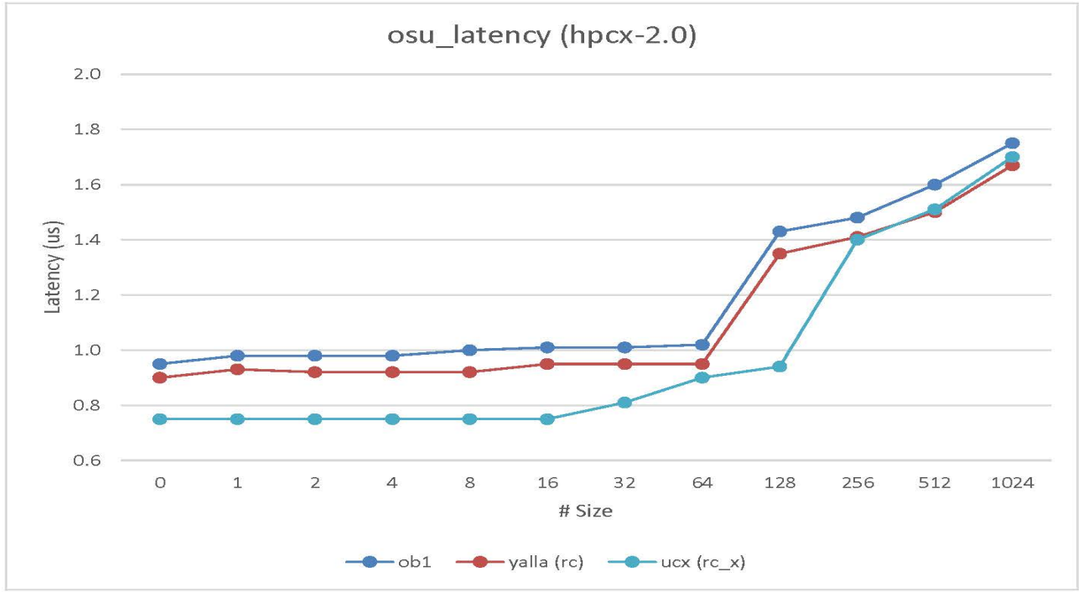
Graph 500 RMA Acceleration

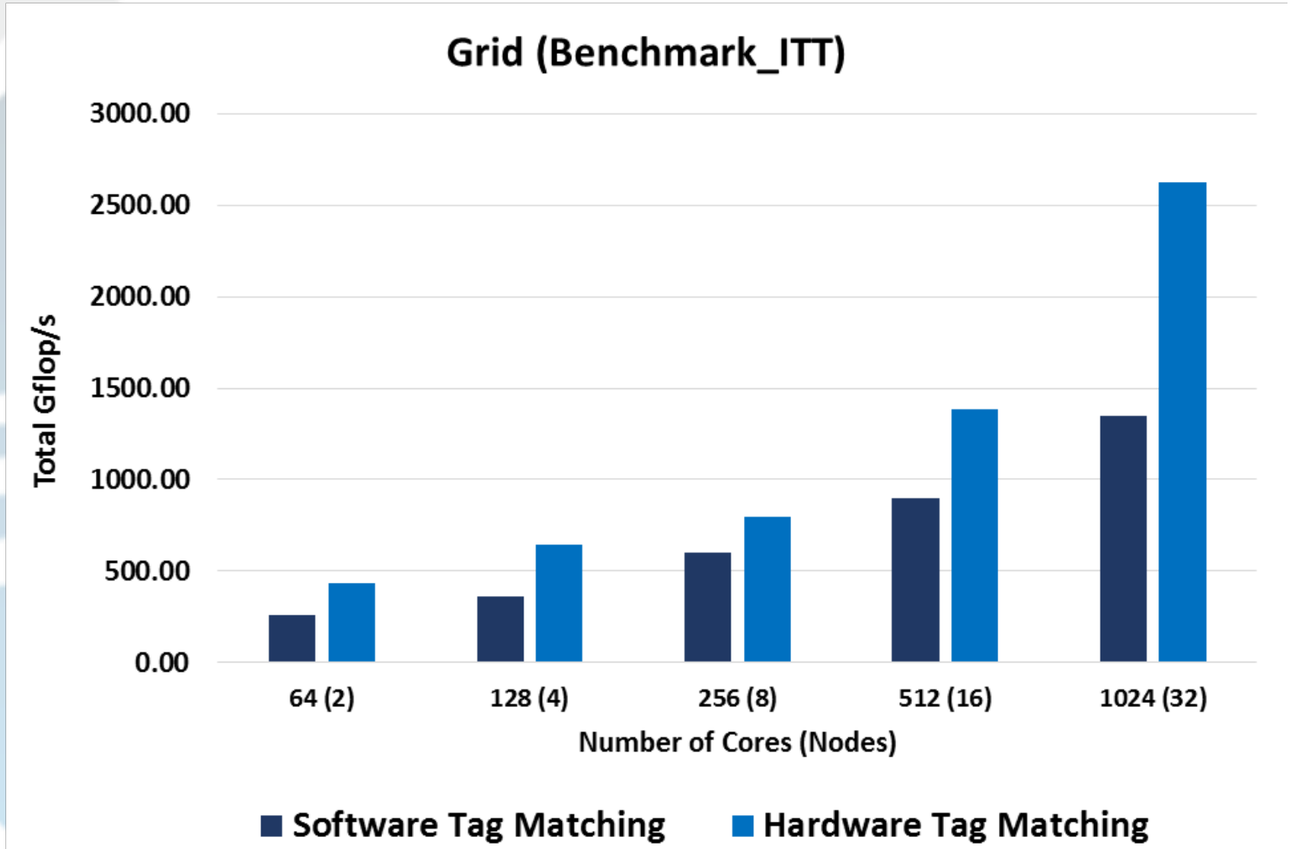


UCX support added

Point-to-Point acceleration library
- Send/Recv
- Put/Get
- Atomics

Optimized for InfiniBand hardware
- Accelerated verbs
- Hardware capabilities

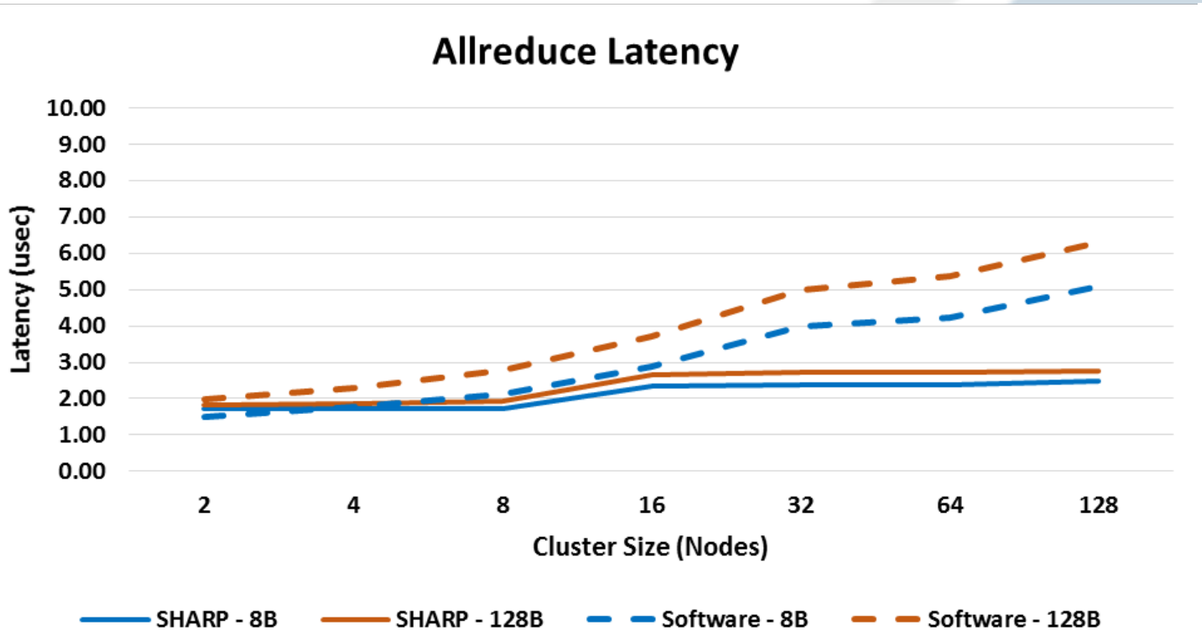


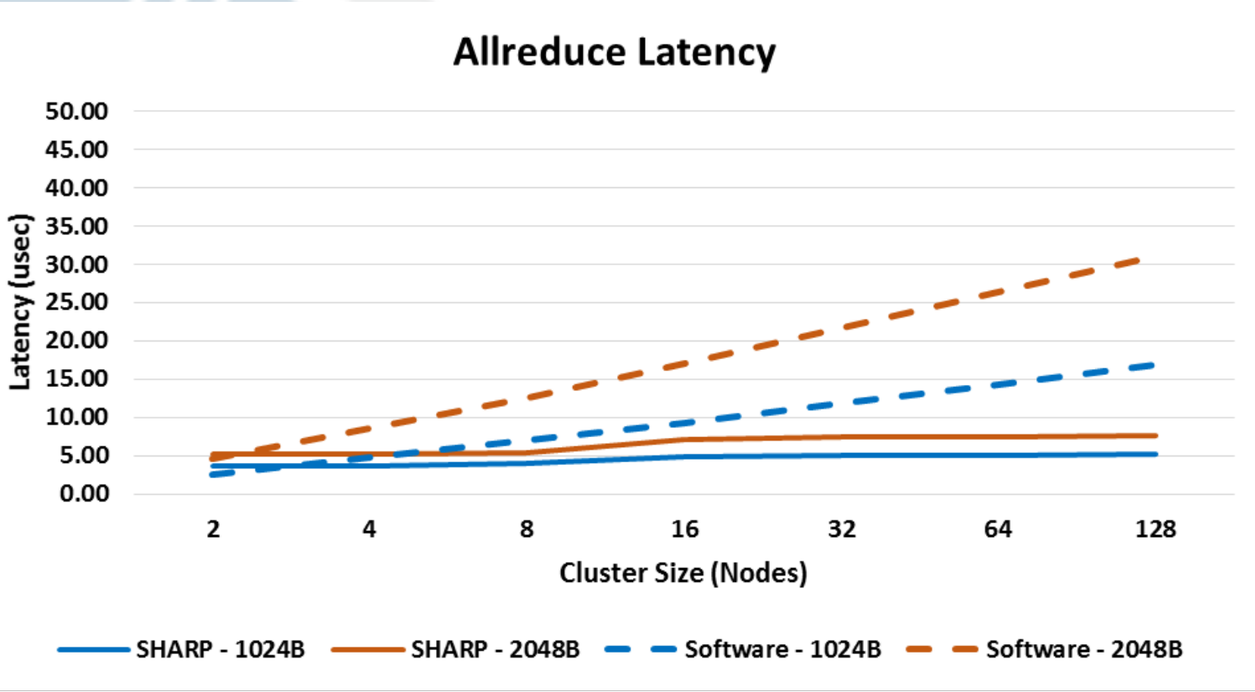


InfiniBand SHARP collectives

Scalable High Performance Collective Offload
- Barrier, Reduce, All-Reduce, Broadcast and more
- Sum, Min, Max, Min-loc, max-loc, OR, XOR, AND
- Integer and Floating-Point, 16/32/64 bits

SHARP
Scalable Hierarchical Aggregation and Reduction Protocol

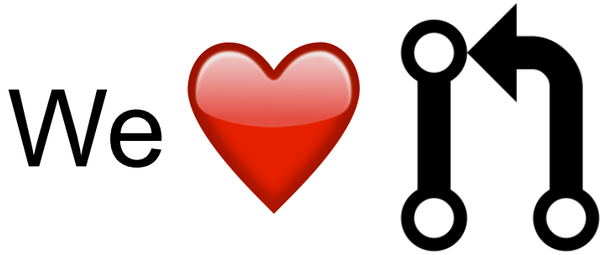




Wrap up

Where do we need help?

- Code
  - Any bug that bothers you
  - Any feature that you can add
- ***User documentation***
- Testing (CI, nightly)
- Usability
- Release engineering



Come join us!

cisco

ICL UT

arm

IBM Spectrum MPI

FUJITSU

RIST

ECP
EXASCALE COMPUTING PROJECT

Los Alamos
NATIONAL LABORATORY
EST. 1943

Mellanox
TECHNOLOGIES

informatics mathematics
Inria