Open MPI for Exascale (OMPI-X)

David E. Bernholdt (ORNL) *for the team*

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

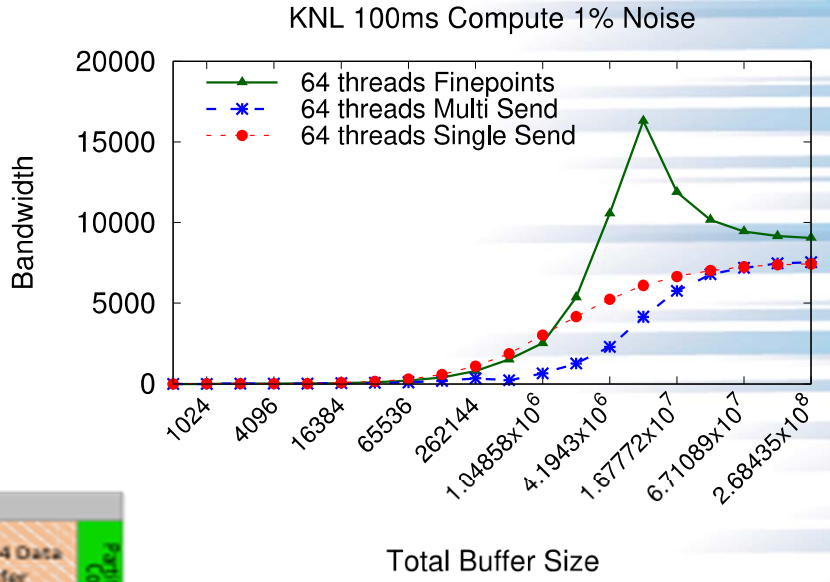
Recent Progress

- 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)

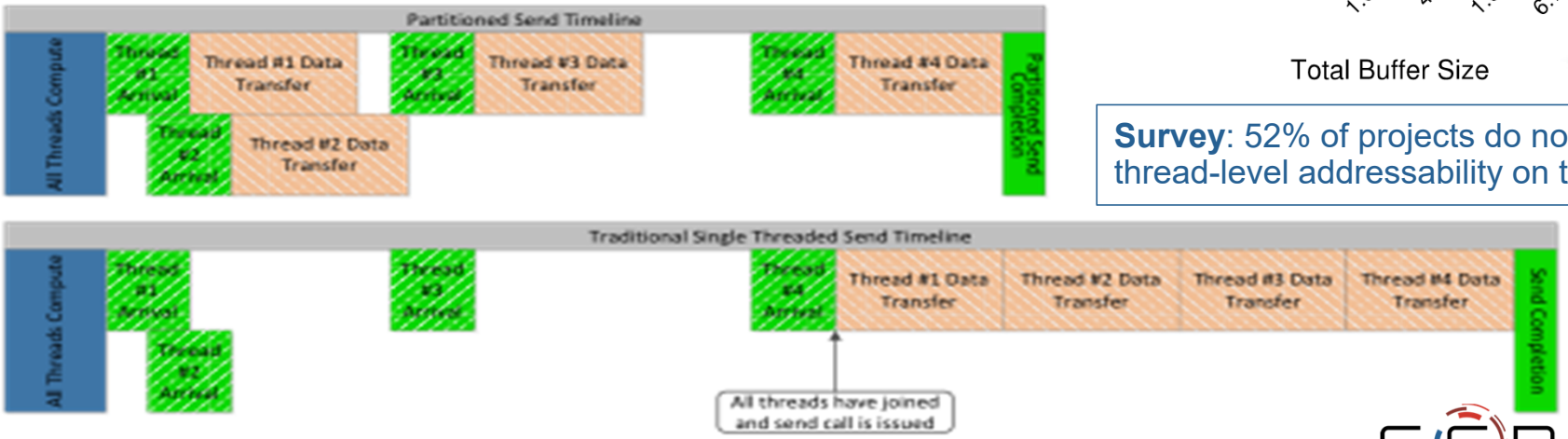
Survey:
- 86% of projects plan to use multiple threads per rank
- 45% use OpenMP; 21% use Kokkos or RAJA

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”



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



Early-bird Communication Example

EXASCALE COMPUTING PROJECT

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.

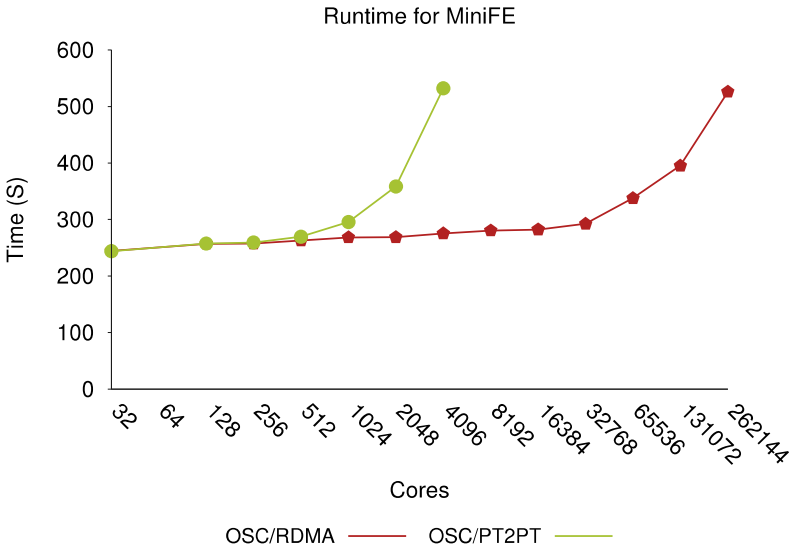
Survey: interest in job-to-job communication capabilities, could be facilitated by Sessions

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

Open MPI 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



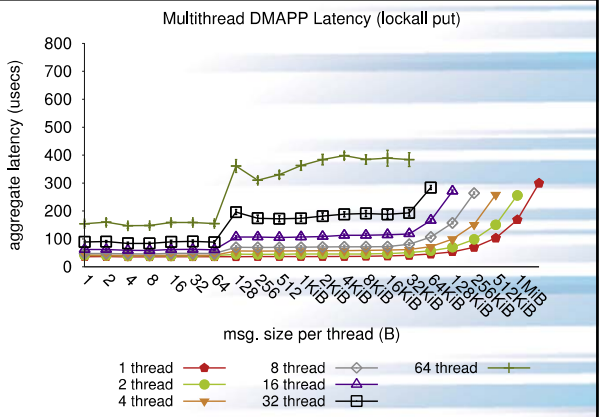


Fig. 1: DMAPP latency put-lockall

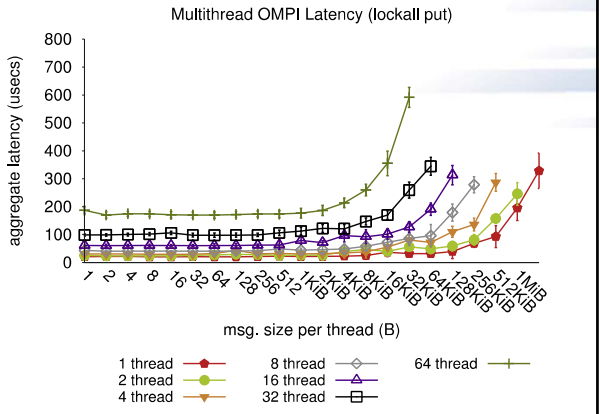
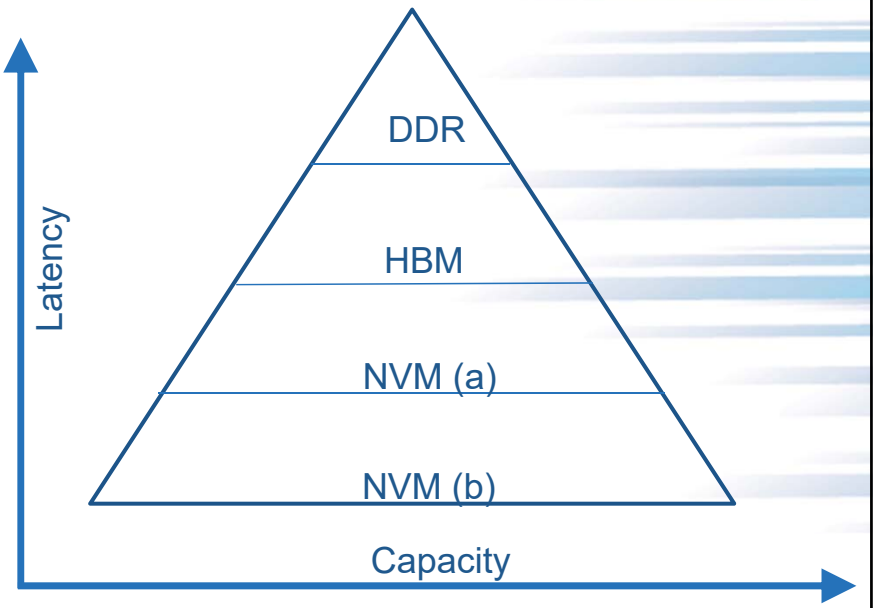


Fig. 2: OMPI latency put-lockall

Complex Memory Hierarchies

Motivation & 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.