Enhancing the MPI Sessions Prototype for Use on Exa-Scale Systems

Tom Herschberg (UTC)
Howard Pritchard (LANL) - mentor

8/6/20

LANL LA-UR-20-25910

Covered today

- MPI Sessions - what is this?
- Open MPI and MPI Sessions Prototype
- Challenges of implementing MPI Sessions, especially when tag matching is done in hardware
- The algorithm used and its performance

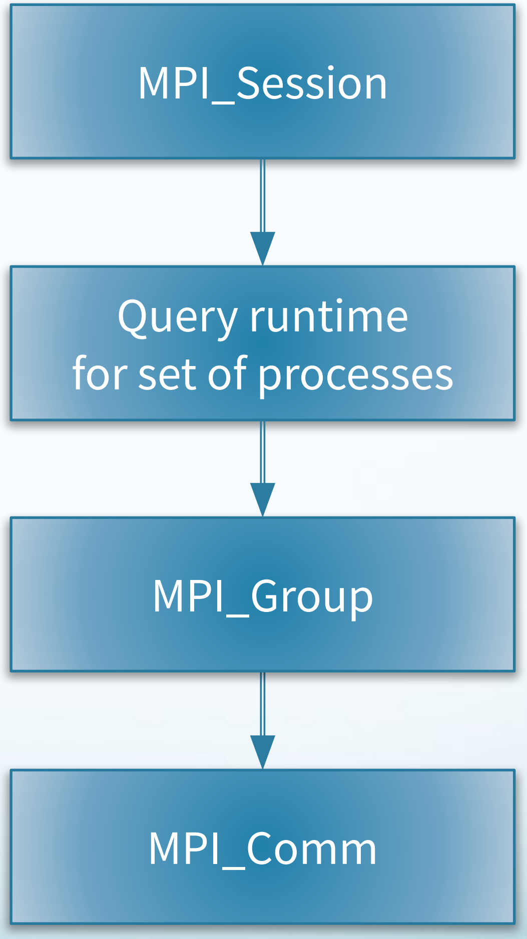
Problems with MPI_Init

- All MPI processes must initialize MPI exactly once
- MPI cannot be initialized within an MPI process from different application components without coordination
- MPI cannot be re-initialized after MPI is finalized

Sessions – a new way to start MPI

- General scheme:
  - Query the underlying run-time system
    - Get a “set” of processes
  - Determine the processes you want
    - Create an MPI_Group
  - Create a communicator with just those processes
    - Create an MPI_Comm

Will be in the MPI 4 standard



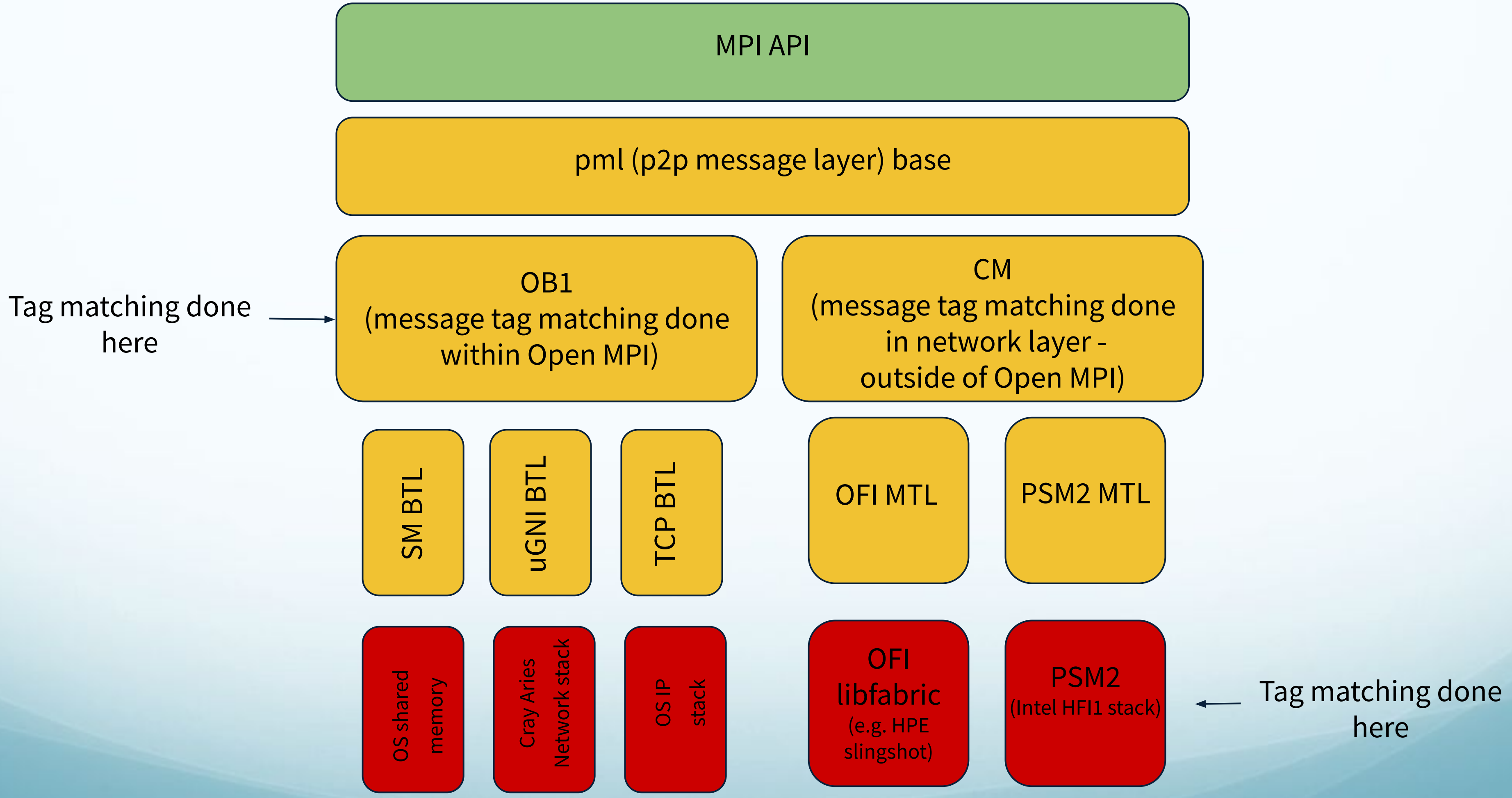
MPI Sessions Prototype

- Implemented in a fork of Open MPI on GitHub
    - https://github.com/hpc/ompi/tree/sessions_new
- Fully functional - implements the MPI Sessions functionality to appear in MPI 4.0 standard
- Uses an extended communicator ID (ExCID) 128 bit structure to support the MPI Sessions function which produces an MPI communicator from a MPI group
- Initial prototype only supported Sessions API for the PML/OB1 messaging component

Challenges Using MPI Sessions

- No parent communicator (MPI_COMM_WORLD) to use for CID generation
- Concept of a 128 bit ExCID was introduced in the prototype to amortize high cost of getting a unique 64 bit number from the runtime system. Upper 64 bits of ExCID are generated based on applications MPI communicator creation pattern
- However, efficient tag matching in software or hardware is best done using smaller quantities (64 bits or smaller)
- Newer networks (HPE slingshot, Nvidia IB nics) support MPI message tag matching in hardware, but need a 64 bit value to do so

Open MPI structure (simplified)



MPI API

pml (p2p message layer) base

Tag matching done here

OB1
(message tag matching done within Open MPI)

CM
(message tag matching done in network layer - outside of Open MPI)

SM BTL

uGNI BTL

TCP BTL

OFI MTL

PSM2 MTL

OS shared memory

Cray Aries Network stack

OS IP stack

OFI libfabric
(e.g. HPE slingshot)

PSM2
(Intel HFI1 stack)

Tag matching done here

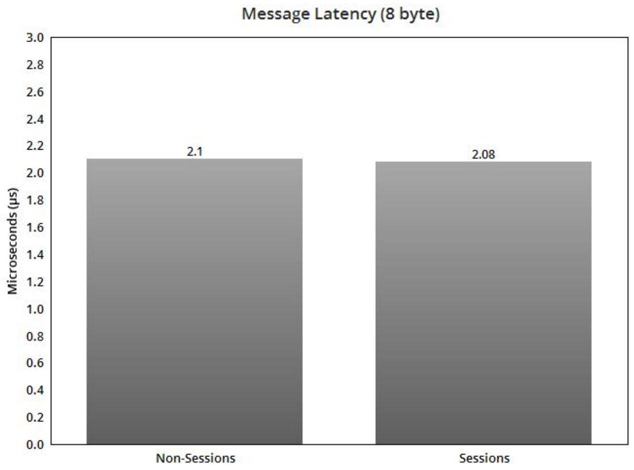
Algorithm

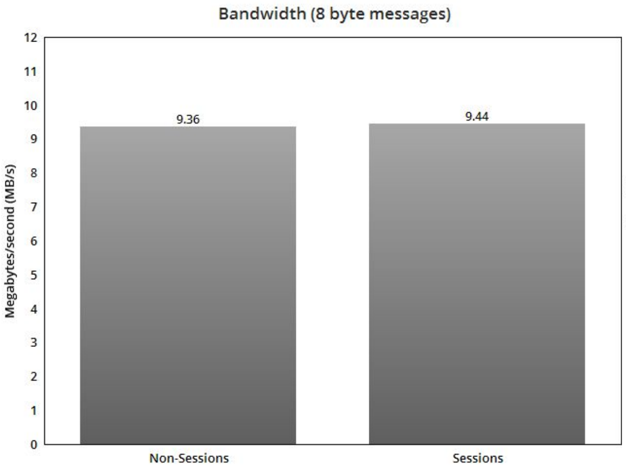
Rank 0 wants to send a message to rank 1, but only has rank 1's ExCID:

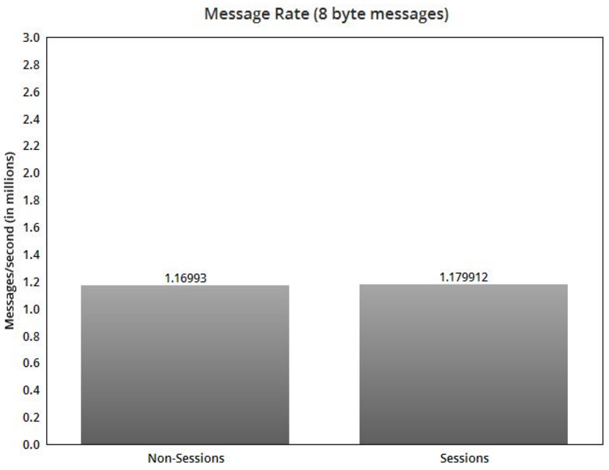
1. Rank 0 sends an untagged control message to Rank 1 containing the ExCID for the current communicator, Rank 0's rank in that communicator, and Rank 0's local 64-bit CID (needed for tag matching).
2. Rank 0 posts a receive buffer and waits for Rank 1's response.
3. Rank 1 posts a receive buffer and receives Rank 0's control message.
4. Rank 1 saves Rank 0's local 64-bit CID, then sends an untagged control message back to Rank 0 containing the ExCID, Rank 1's rank, and Rank 1's local 64-bit CID.
5. Rank 0 receives Rank 1's response and saves Rank 1's local 64-bit CID.
6. Rank 0 and Rank 1 can now exchange tagged messages normally using each other's local CID without needing to send control messages back and forth.

Performance

- Tests from the Ohio State University Microbenchmarks test suite were modified to use Sessions functions in order to study their performance
- There were negligible performance differences between the non-Sessions and Sessions tests for message latency, bandwidth, and messages/second







Funding Acknowledgments

ECP

EXASCALE COMPUTING PROJECT

DEPARTMENT OF ENERGY

UNITED STATES OF AMERICA

References

- https://doi.org/10.1109/CLUSTER.2019.8891002 (paper describing prototype and exCID algorithm)
- OFI Libfabric (https://ofiwg.github.io/libfabric/)
- Open MPI (https://www.open-mpi.org/)

Backup stuff

Future work

- Address potential scalability issues (both on PMIx and OMPI sides)
  - Procs arg to PMIx_Group_construct (PMIx)
  - Per OMPI proc memory needed for extended CID handling (OMPI)
- Procs need to be associated with multiple PMIX_PSET_NAMEs (PMIx)
- Enhance mechanism for creating PMIx PSETs (PMIx)
- Handling (unexpected) process exit (OMPI)
- Group expansion (PMIx)
- Investigate use of Sessions in various workflows (tried DASK) (OMPI/PMIx)