Research Infrastructure

Derek Schafer (dschafer1@unm.edu)

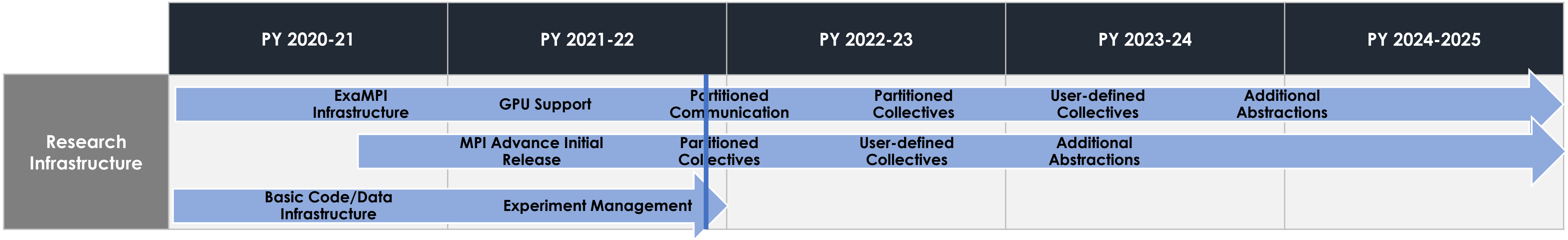
University of New Mexico

Center for Advanced Research Computing

September 29th, 2022

Overview

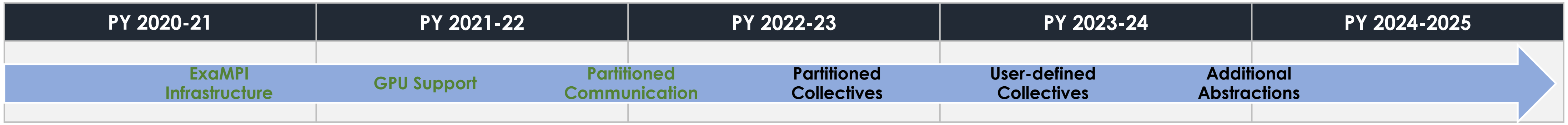
- Efforts focused into three areas:
  - MPI Advance
  - ExaMPI
  - Experiment Management
- Timeline:



ExaMPI

- Quick recap:
  - Modern C++ MPI implementation
  - Features strong progress, most of the common MPI 3.1 functions
  - Designed to for experimentation within MPI implementations
- Previous year’s goals are complete
- Publication:

D. Schafer, T. Hines, E. D. Suggs, M. Rüfenacht and A. Skjellum, "Overlapping Communication and Computation with ExaMPI's Strong Progress and Modern C++ Design," 2021 Workshop on Exascale MPI (ExaMPI), 2021, pp. 18-26



PY 2020-21 | PY 2021-22 | PY 2022-23 | PY 2023-24 | PY 2024-2025

ExaMPI Infrastructure → GPU Support → Partitioned Communication → Partitioned Collectives → User-defined Collectives → Additional Abstractions

ExaMPI – Accomplished Milestones

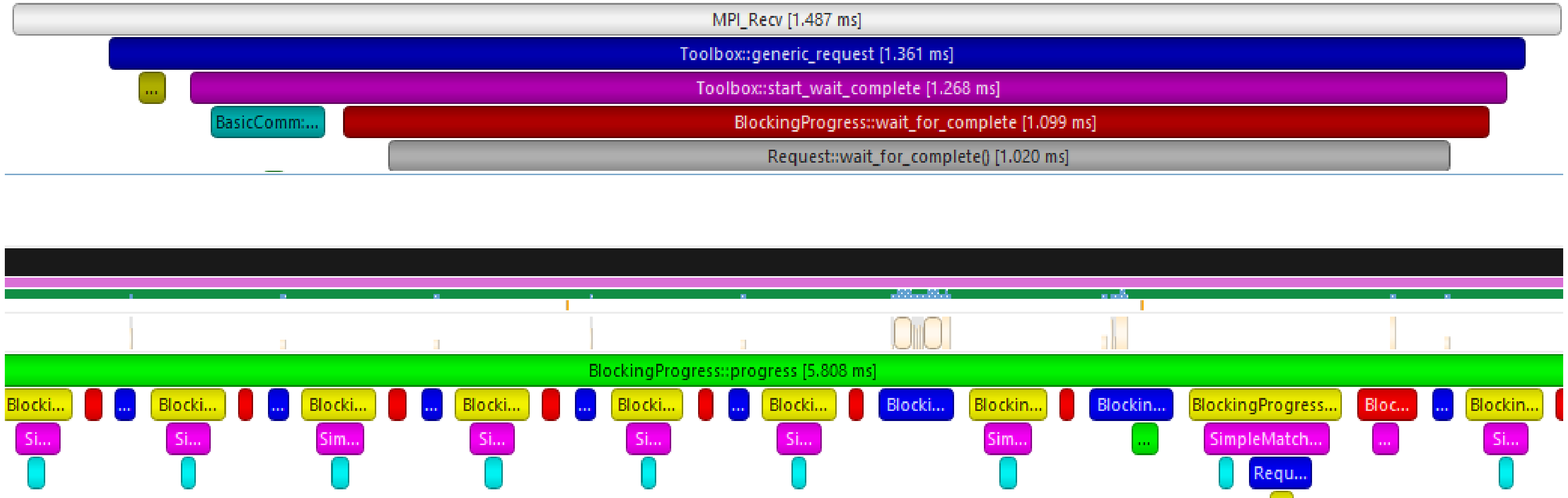
- Added basic GPU support
  - GPU Direct not supported yet
  - Each message has a packing buffer (internally)
  - Transports can query type of buffer, ask to pack message
- Added basic partitioned P2P
  - Current algorithm has the sender just report the number of partitions
  - Also supports MPIPCL library

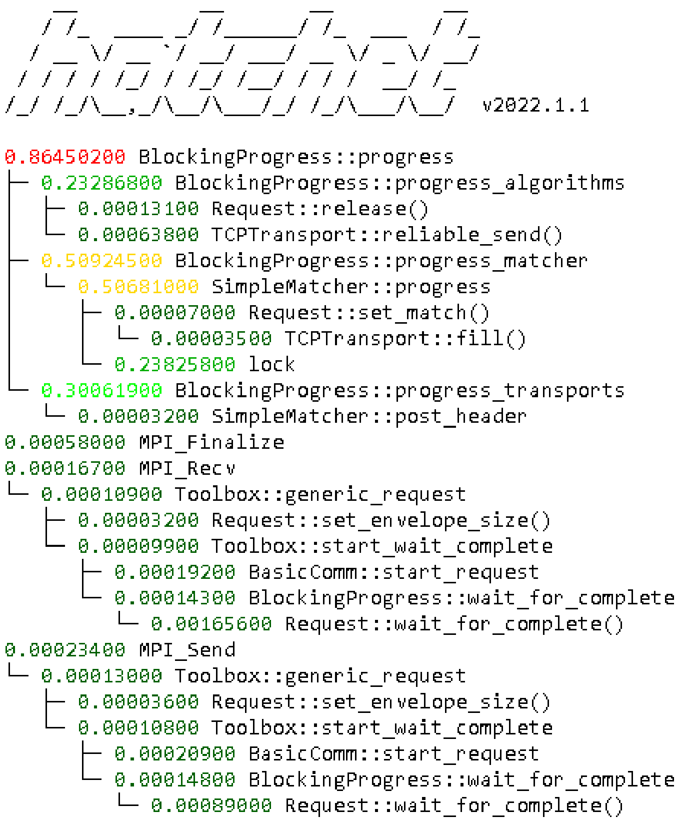
ExaMPI – Other minor features

- Various MPI functions added:
  - Strengthened MPI Datatype support (Struct types, hvectors, some edge cases)
  - Added collective variants (Allgatherv, Alltoallv/w, Gatherv, Scatterv)
  - Miscellaneous functions and constants (i.e., MPI_Cart_sub, MPI_Probe, MPI_Bottom, MPI_AINT)
- Tightened up:
  - Compiler wrappers (mpicc, etc – mpifort soon)
  - Compiler support (Clang)
  - CMake detection of ExaMPI
- Added support for LLNL's lrun job system on Lassen
- More transports and use of dynamic connection forming
- Added config file to specify transports, progress options at runtime

Profiling ExaMPI

- Using Caliper to instrument ExaMPI
- Can turn export to Hatchet or NVTX for analysis
- Support for message tracing


MPI_Recv [1.487 ms]
Toolbox::generic_request [1.361 ms]
...
Toolbox::start_wait_complete [1.268 ms]
BasicComm:...
BlockingProgress::wait_for_complete [1.099 ms]
Request::wait_for_complete() [1.020 ms]
BlockingProgress::progress [5.808 ms]
Blocki...
...
Si...
Sim...
Blockin...
BlockingProgress...
Bloc...
SimpleMatch...
Requ...

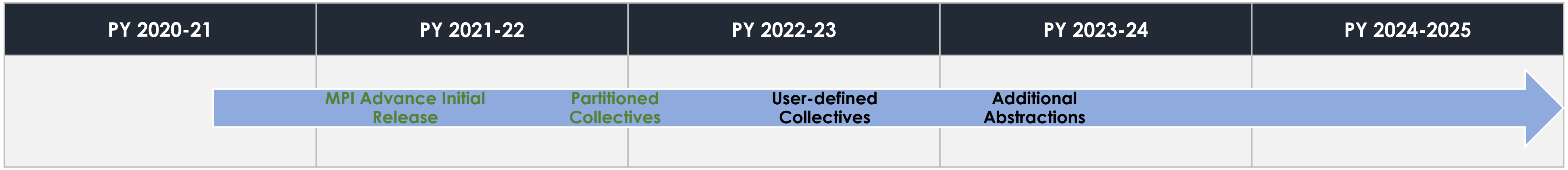

```
    __          __       __         __
   / /_  ____ _/ /______/ /_  ___  / /_
  / __ \/ __ `/ __/ ___/ __ \/ _ \/ __/
 / / / / /_/ / /_/ /__/ / / /  __/ /_
/_/ /_/\__,_/\__/\___/_/ /_/\___/\__/  v2022.1.1

0.86450200 BlockingProgress::progress
├─ 0.23286800 BlockingProgress::progress_algorithms
│  ├─ 0.00013100 Request::release()
│  └─ 0.00063800 TCPTransport::reliable_send()
├─ 0.50924500 BlockingProgress::progress_matcher
│  └─ 0.50681000 SimpleMatcher::progress
│     ├─ 0.00007000 Request::set_match()
│     │  └─ 0.00003500 TCPTransport::fill()
│     └─ 0.23825800 lock
└─ 0.30061900 BlockingProgress::progress_transports
   └─ 0.00003200 SimpleMatcher::post_header
0.00058000 MPI_Finalize
0.00016700 MPI_Recv
└─ 0.00010900 Toolbox::generic_request
   ├─ 0.00003200 Request::set_envelope_size()
   └─ 0.00009900 Toolbox::start_wait_complete
      ├─ 0.00019200 BasicComm::start_request
      └─ 0.00014300 BlockingProgress::wait_for_complete
         └─ 0.00165600 Request::wait_for_complete()
0.00023400 MPI_Send
└─ 0.00013000 Toolbox::generic_request
   ├─ 0.00003600 Request::set_envelope_size()
   └─ 0.00010800 Toolbox::start_wait_complete
      ├─ 0.00020900 BasicComm::start_request
      └─ 0.00014800 BlockingProgress::wait_for_complete
         └─ 0.00089000 Request::wait_for_complete()
```

What’s Next for ExaMPI

- Developmental Areas:
  - Fortran support that is needed for some applications
  - MPI File support (possibly through ROMIO library)
  - Other miscellaneous MPI functions that are used by a given application
  - Other network transports (gpu-direct, ucx, etc)
- Research Areas:
  - Measure performance with other benchmarks, applications
  - Adding partitioned collectives to ExaMPI (and/or supporting MPIPCL version)
  - Continue improving message tracing capabilities
  - GPU performance with datatypes, partitioned communication

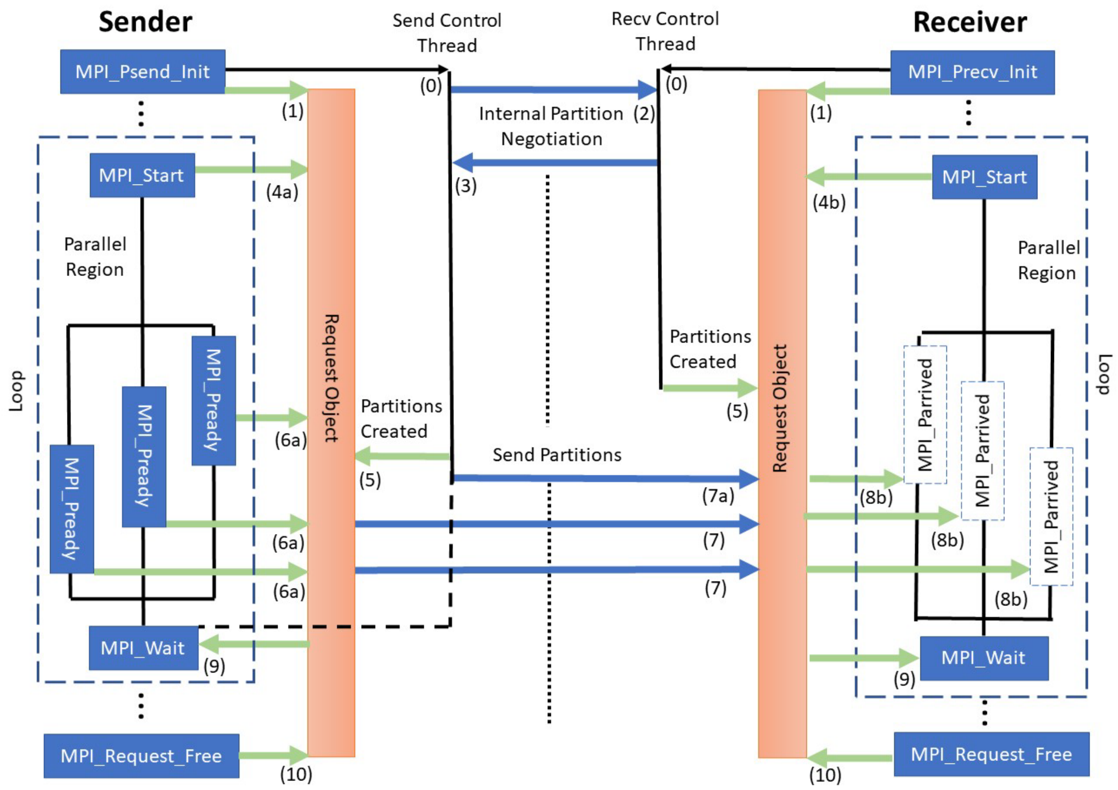
MPI Advance

- A collection of MPI libraries showcasing new APIs or optimizations of current MPI APIs
- GitHub organization
- Current libraries:
  - MPIPCL
  - Locality Aware MPI
- MPIPCL successfully used in EuroMPI2022 Tutorials



MPIPCL

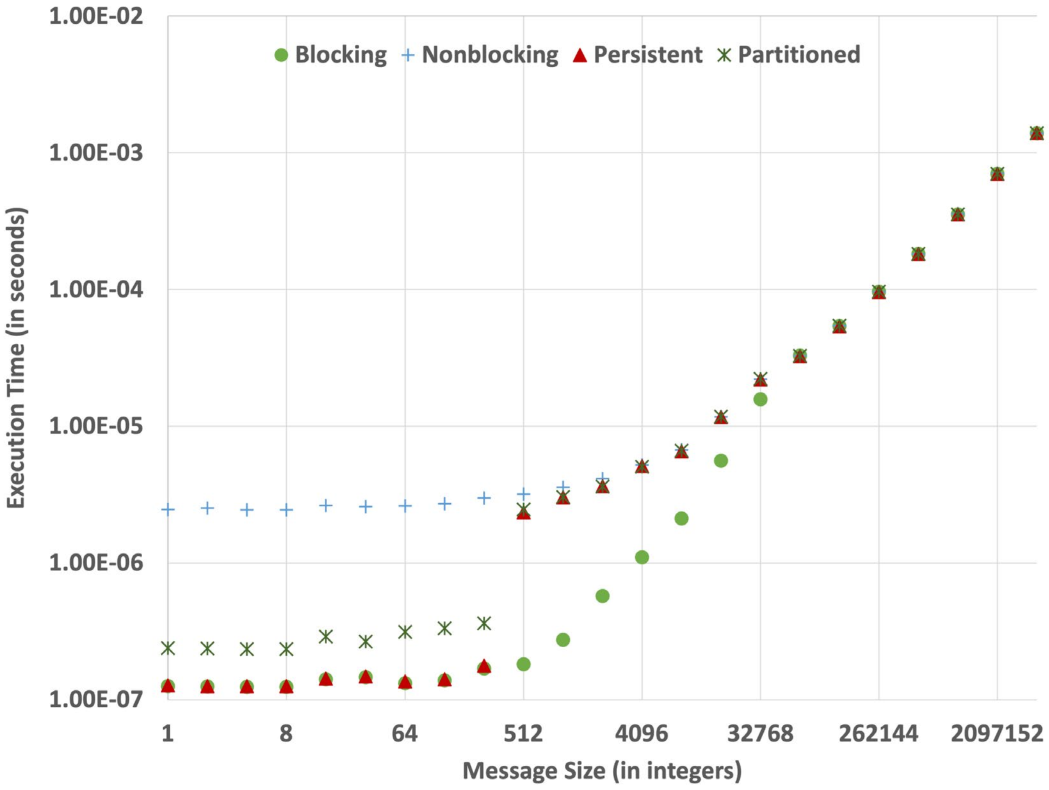
- Implements all MPI 4.0 partitioned communication APIs
- Is a layered library on top of existing MPI implementations
- Technical details:
    - Uses MPI Persistent P2P APIs
    - Has a progress thread for partition negotiation
    - Requires custom start/wait/test APIs



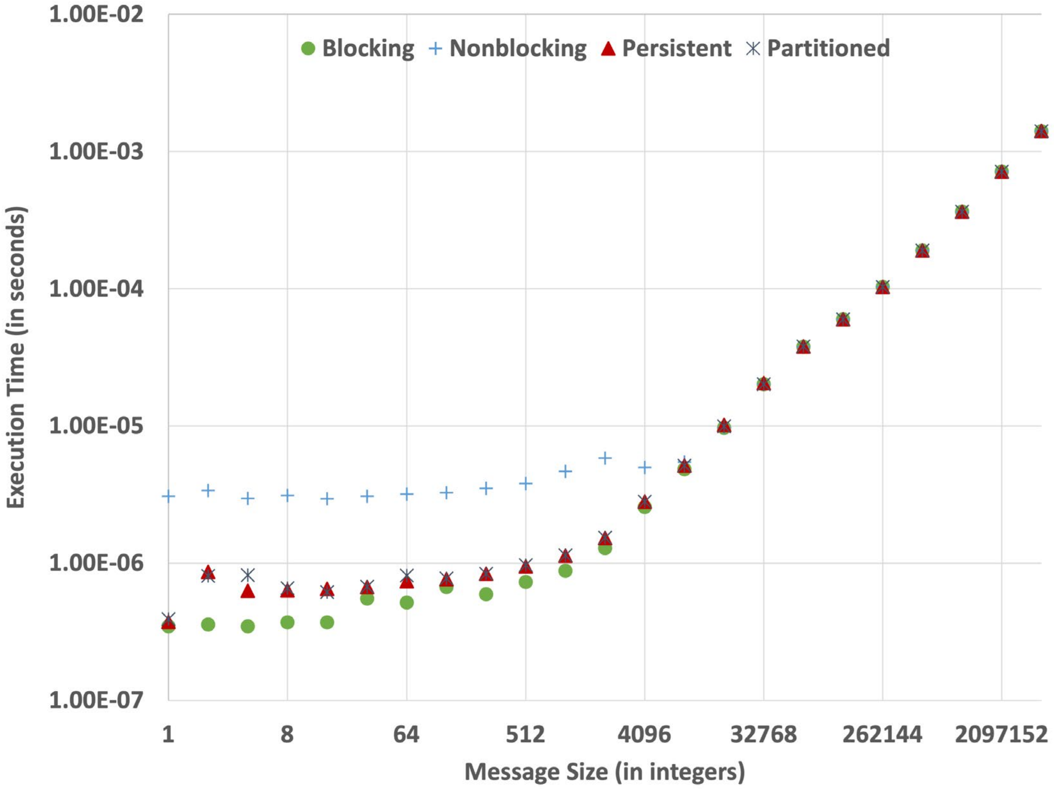
Architectural Overview of MPIPCL

M. G.F. Dosanjh, A. Worley, D. Schafer, P. Soundararajan, S. Ghafoor, A. Skjellum, P. V. Bangalore, R. E. Grant, Implementation and evaluation of MPI 4.0 partitioned communication libraries, Parallel Computing, Volume 108, 2021, https://doi.org/10.1016/j.parco.2021.102827.

Partitioned Bindings Testing

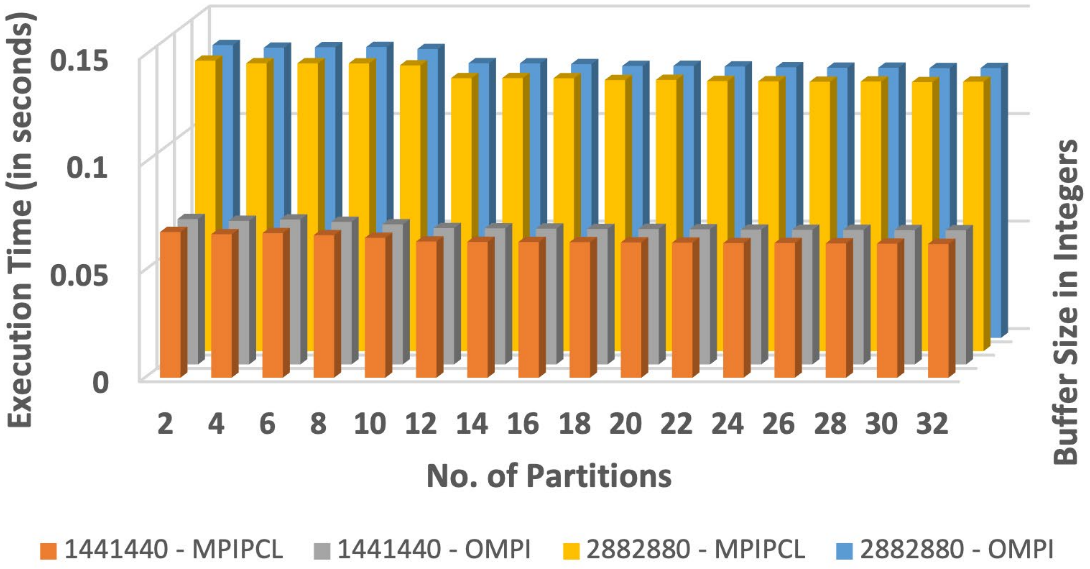


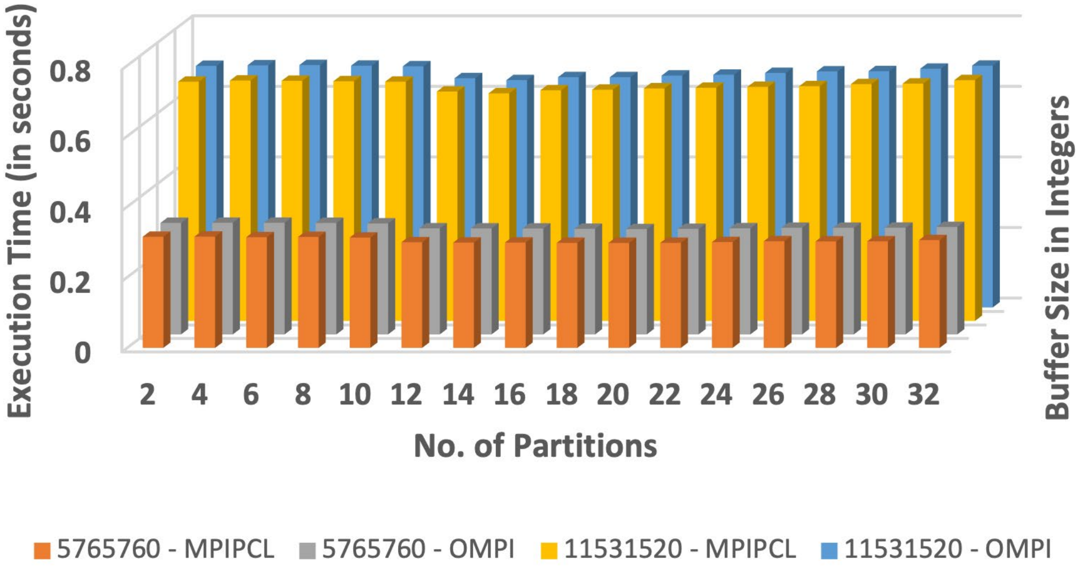
OpenMPI



IntelMPI

MPIPCL vs. OpenMPI Implementation





- Due to scale of timings, the results are split into two graphs
- The graphs showcase **different** message sizes

Locality Aware MPI

- Locality-Aware Persistent Neighborhood collectives
    - Neighbor Alltoallv, Neighbor alltoallw
    - Requires use of special topology communication
    - Integrated into Hypre (see Gerald's talk)
- Locality-Aware Collectives: Allgather, Alltoall, Alltoallv
- Uses MPI Profiling library to hook into MPI
- Allows for optimizations within existing codebases with minimal changes to existing code

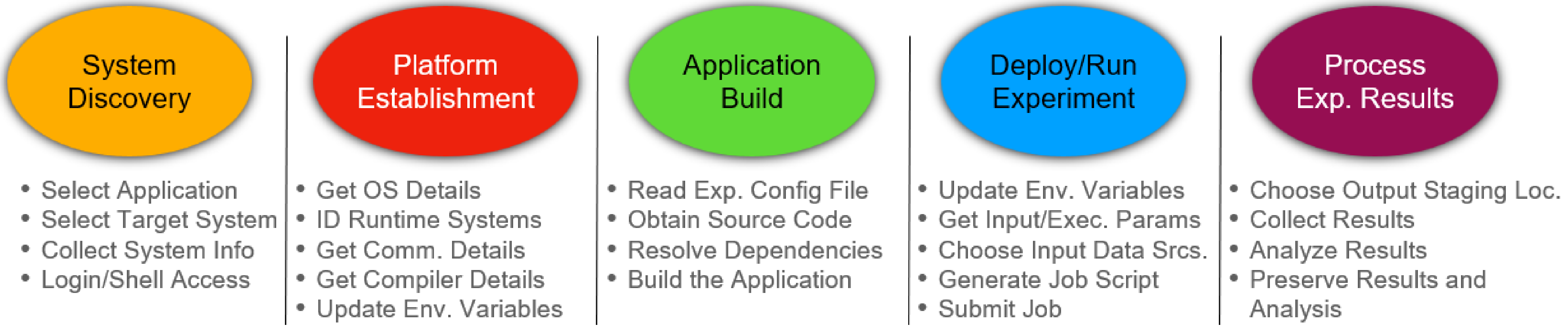
Bienz A, Gropp WD, Olson LN. Reducing communication in algebraic multigrid with multi-step node aware communication. *The International Journal of High Performance Computing Applications*. 2020;34(5):547-561.

MPI Advance Next Steps

- MPIPCL – Partitioned Collectives:
  - D. Holmes, et al., "Partitioned Collective Communication," in 2021 Workshop on Exascale MPI (ExaMPI), St. Louis, MO, USA, 2021 pp. 9-17.)
  - First implementation in progress with collaboration from TN Tech
- User-defined collectives
- GPU triggered communication abstractions
- Potential integration of MPI Advance libraries
  - In other software packages, applications
  - Or our own bundle of libraries

Unified Lab Notes Framework

- Proposes an experiment management framework for large-scale HPC systems
- Enhances productivity for the research team
- Promotes experimental integrity and reproducibility
- Provides minimal infrastructure for greater flexibility



System Discovery

- Select Application
- Select Target System
- Collect System Info
- Login/Shell Access

Platform Establishment

- Get OS Details
- ID Runtime Systems
- Get Comm. Details
- Get Compiler Details
- Update Env. Variables

Application Build

- Read Exp. Config File
- Obtain Source Code
- Resolve Dependencies
- Build the Application

Deploy/Run Experiment

- Update Env. Variables
- Get Input/Exec. Params
- Choose Input Data Srcs.
- Generate Job Script
- Submit Job

Process Exp. Results

- Choose Output Staging Loc.
- Collect Results
- Analyze Results
- Preserve Results and Analysis

Thank you!

Any questions?