A Lightweight Communication Interface for Asynchronous Many-Task Systems

Omri Mor[1], Jiakun Yan[1], Hartmut Kaiser[2], Marc Snir[1]

[1]Department of Computer Science, University of Illinois Urbana-Champaign

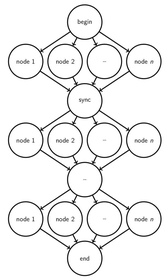
[2]Center of Computation and Technology, Louisiana State University

Motivation

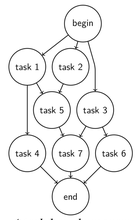
Modern Parallel Architecture:
- Increased intra-node parallelism
- Increased heterogeneity
- Improved networking hardware

Asynchronous Many-Task (AMT) Model:
- Algorithms decomposed into **tasks** & **dependencies**
- Smart runtime handles **mapping**, **scheduling,** and **data movement** of tasks onto compute resources
- Examples: **HPX**, **PaRSEC**, Legion, StarPU
- Communication layer: usually MPI or GASNet

New Communication Paradigm:
- Multithreaded
- Irregular destinations
- Message classes: control & data
- Communication priorities
- Nondeterministic, execution-based ordering



A BSP program

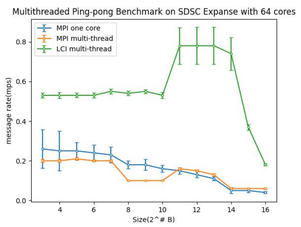


A task-based program

Lightweight Communication Interface (LCI) [2]

Low-level Communication Library:
- Intended user: high-level library developers
- Similar to UCX, libfabric, and GASNet, as opposed to MPI

Designed for AMT Runtimes:
- Applies to other irregular applications, such as graph analytics or sparse linear algebra
- Easily modifiable research library, for studying communication API design and implementation



Major Features:
- **Multithreaded performance as top priority**
- **Versatile communication interface**
- **Explicit control of communication behaviors and resources**

HPX [3] and PaRSEC [1]: New LCI Backends [4, 5]

High Performance ParalleX (HPX):
- Implementation of ParalleX execution model, standards-focused C++ API
- Asynchronous task execution and arbitrary task graph construction.
- Developed by the STE||AR Group

Parallel Runtime Scheduling and Execution Controller (PaRSEC):
- Generic framework for executing distributed task-based applications
- Support for multiple Domain-Specific Language backends
- Developed by the Innovative Computing Laboratory (ICL) at the University of Tennessee, Knoxville

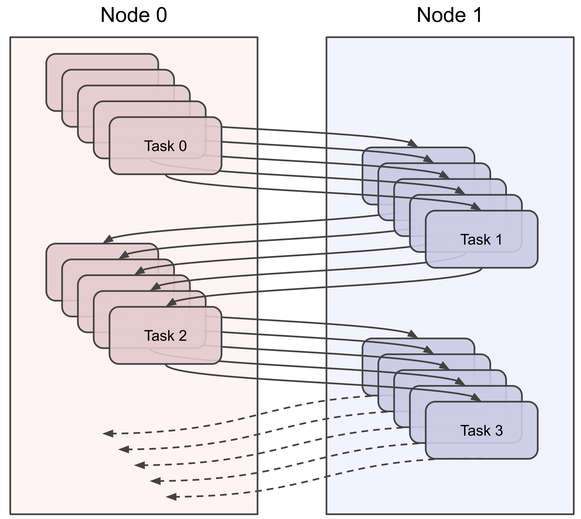
Task-based Multi-Chain Ping-Pong

Motivation:
- Communication layer benchmarks necessary to measure and improve efficiency
- Traditional communication benchmarks do not directly apply to AMT runtimes

Design:
- **"Chains" of tasks** spawned on two nodes and alternating between nodes
- Task chains associated with **data movement**
- Tasks optionally execute **computation** on data

Variables:
- **Message size**, **chain count,** chain length, and **task granularity**

Measurement:
- Basic communication characteristics: latency, message rate, and bandwidth
- Realistic cache effects with task modification of communication buffers
- Communication-communication and computation-communication overlap



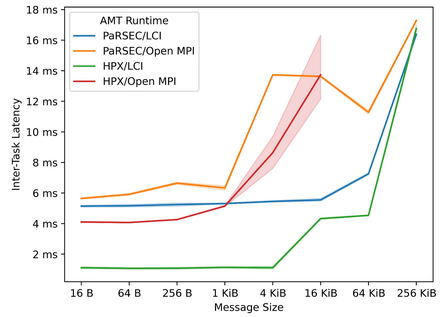
References

1. George Bosilca et al. "DAGuE: A generic distributed DAG engine for High Performance Computing". Parallel Computing. 38, 1. 2012. (https://github.com/icldisco/parsec)
2. Hoang-Vu Dang, Marc Snir, and William Gropp. "Towards millions of communicating threads." Proceedings of the 23rd European MPI Users' Group Meeting. 2016. (https://github.com/uiuc-hpc/LC)
3. Hartmut Kaiser et al. "HPX: A task based programming model in a global address space." Proceedings of the 8th International Conference on Partitioned Global Address Space Programming Models. 2014. (https://github.com/STEllAR-GROUP/hpx)
4. Omri Mor, George Bosilca, and Marc Snir. "Improving the Scaling of an Asynchronous Many-Task Runtime with a Lightweight Communication Engine." Proceedings of the 52nd International Conference on Parallel Processing. 2023.
5. Jiakun Yan, Hartmut Kaiser, and Marc Snir. "Design and Analysis of the Network Software Stack of an Asynchronous Many-task System--The LCI parcelport of HPX." Proceedings of the SC'23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis. 2023.

Evaluation

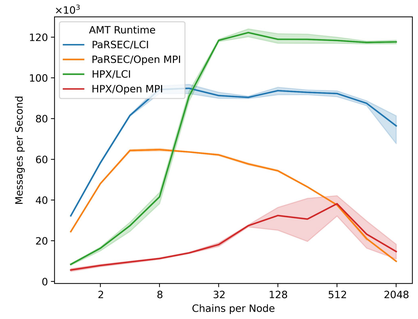
Execution Environment:
- Expanse cluster at San Diego Supercomputer Center
- 2× AMD EPYC 7742: 128 cores @ 2.25 GHz with 256 GiB DDR4
- Mellanox ConnectX-6: 2× HDR InfiniBand links in a Hybrid Fat-Tree topology
- Rocky Linux 8.8 with Linux 4.18.0-477.15.1
- Open MPI 4.1.5 with UCX 1.14.0, using AMD Optimizing C/C++ Compiler

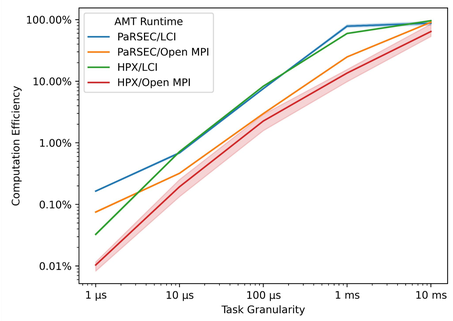
Results:



Communication Latency
- Parameters:
  - # chains: 512 per node
  - **Message size**: 16 B–256 KiB
- Measurement:
  - Mean time between task executions, driven by **communication latency**
- LCI reduces latency in both runtimes, with significant impact on HPX performance

Message Rate
- Parameters:
  - Message size: 16 KiB
  - **# chains:** 1–2048 per node
- Measurement: **message rate**
- Using LCI greatly improves the maximum message rates of both PaRSEC and HPX
- MPI backends for both runtimes suffer degraded performance when many messages are communicated simultaneously





Computation Efficiency
- Parameters:
  - **Task size**: 1 µs–10 ms
  - Message size: 16 KiB
  - # chains: 512 per node
- Measurement:
  - **Computation efficiency**
- Both runtimes are able to overlap computations and communications more efficiently when using LCI

NSF

ILLINOIS

A Lightweight Communication Interface for Asynchronous Many-Task Systems

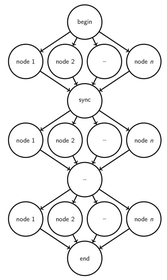
Omri Mor, Jiakun Yan, and Marc Snir
Department of Computer Science, University of Illinois Urbana-Champaign

Motivation

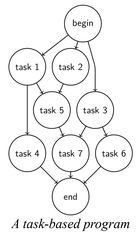
Modern Parallel Architecture:
- Increased intra-node parallelism
- Increased heterogeneity
- Programmable networking

Asynchronous Many-Task (AMT) Model:
- Algorithms decomposed into **tasks** & **dependencies**
- Smart runtime handles **mapping**, **scheduling,** and **data movement** of tasks onto compute resources
- Examples: **HPX**, **PaRSEC**, Legion, StarPU
- Communication layer: usually MPI or GASNet

New Communication Paradigm:
- Multithreaded
- Irregular destinations
- Message classes: control & data
- Communication priorities
- Nondeterministic, execution-based ordering



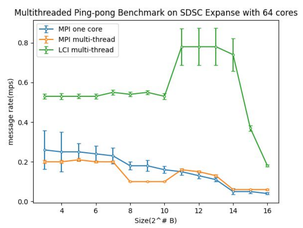
A BSP program



A task-based program

Lightweight Communication Interface (LCI) [1]

Low-level Communication Library:
- Intended user: high-level library developers
- Similar to UCX or libfabric, as opposed to MPI or GASNet



Designed for AMT Runtimes:
- Applies to other irregular applications, such as graph analytics or sparse linear algebra
- Easily modifiable research library, for studying communication API design and implementation

Major Features:

- **Flexible communication primitives and signaling mechanisms**
- **Better multithreaded performance**
- **Explicit runtime control of communication behaviors and resources**

HPX [2] and PaRSEC [3]

High Performance ParalleX (HPX):
- Implementation of ParalleX execution model
- Standards-focused C++ API
- Developed by the STE||AR Group

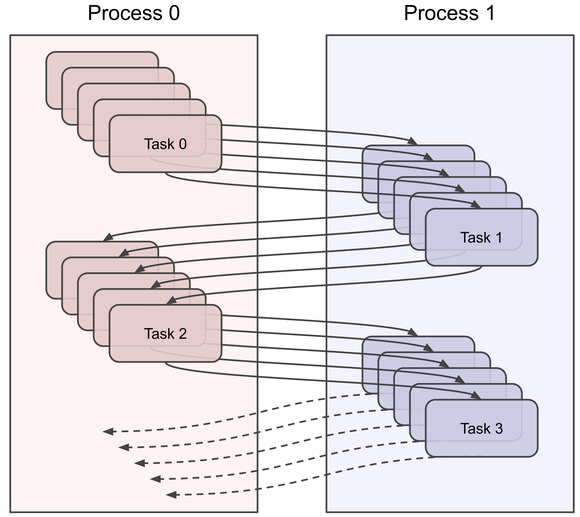
Parallel Runtime Scheduling and Execution Controller (PaRSEC):
- Generic framework for executing distributed task-based applications
- Support for multiple Domain-Specific Language backends
- Developed by the Innovative Computing Laboratory (ICL) at the University of Tennessee, Knoxville
-

Task-based Multi-Chain Ping-Pong

Motivation:
- Communication layer benchmarks necessary to measure and improve efficiency
- Traditional communication benchmarks do not apply to task-based runtimes
- Cross-runtime comparisons enable identification of implementation decisions

Design:
- Serial "chain" of tasks, located on different processes
- Tasks associated with different data, mandating communication
- Multiple sets of tasks approximate traditional benchmark "window size"
- Tasks optionally execute operations on data

Features:
- Variable **message size**, **chain count**, and chain length
- Measure compute/communication overlap with adjustable **compute intensity**
- **Bidirectional** measurement by round-robin instantiation of multiple independent chains on different initial processes
- Comprehensive output statistics for thorough performance characterization



References

1. Hoang-Vu Dang, Marc Snir, and William Gropp. "Towards millions of communicating threads." Proceedings of the 23rd European MPI Users' Group Meeting. 2016. (https://github.com/uiuc-hpc/LC/tree/dev-v1.7)
2. Hartmut Kaiser et al. "HPX: A task based programming model in a global address space." Proceedings of the 8th International Conference on Partitioned Global Address Space Programming Models. 2014. (https://github.com/STEllAR-GROUP/hpx)
3. George Bosilca et al. "DAGuE: A generic distributed DAG engine for High Performance Computing". Parallel Computing. 38, 1. 2012. (https://github.com/icldisco/parsec)
4.

Evaluation

Execution Environment:
- Expanse cluster at San Diego Supercomputer Center
- 2× AMD EPYC 7742: 128 cores @ 2.25 GHz with 256 GiB DDR4
- Mellanox ConnectX-6: 2× HDR InfiniBand links in a Hybrid Fat-Tree topology
- Rocky Linux 8.8 with Linux 4.18.0-477.15.1

Experiments:
- Vary message size, from 8 B to 256 KiB
- Vary number of chains, from 256 to 4096
- Vary compute intensity: 0µs, 10µs, 100µs, and 1000µs task length

Results:
-

NSF
ILLINOIS

Efficient Message Passing Support for Asynchronous Many-Task Systems

Jiakun Yan[1], Hartmut Kaiser[2], Marc Snir[1]

[1]Department of Computer Science, University of Illinois Urbana-Champaign

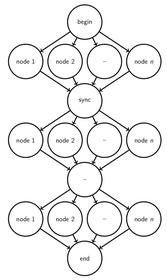
[2]Center of Computation and Technology, Louisiana State University

Motivation

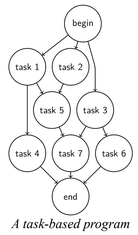
Modern Parallel Architecture:
- Increased intra-node parallelism.
- Increased heterogeneity.
- Powerful Interconnect.

Task-based Programming Model:
- Programmers decompose their programs into **tasks** along with their **dependencies**.
- The runtime will handle the **mapping**, **scheduling,** and **data movement**.
- E.g. HPX, Legion, PaRSEC.
- Communication layer: MPI/GASNet.

New Communication Pattern:
- Multithreaded.
- Irregular destinations.
- Small messages.



A BSP program

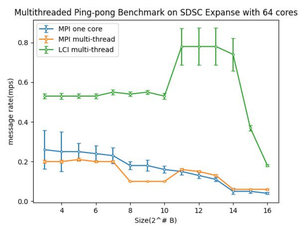


A task-based program

Lightweight Communication Interface (LCI) [1]

A low-level communication library.
- Intended user: high-level library developers.
- Like UCX/Libfabric/GASNet, as opposed to MPI.

Designed with task-based runtime as target clients.
- Applies to other irregular applications such as graph analysis/sparse linear algebra.



Major features:
- **Flexible communication primitives and signaling mechanisms.**
- **Better multithreaded performance.**
- **Explicit user control of communication behaviors and resources.**

HPX [2]

Task-based programming model with implicit task dependency graphs.
- Users invoke tasks like sequential code, and the runtime analyzes data usage and builds a task dependency graph.

Its communication layer implements a "parcelport" interface.
- A parcel consists of a small buffer (control data + small arguments) and optionally a few large zero-copy buffers.
- Current implementations: TCP, MPI, libfabric (work in progress).

The LCI Parcelport for HPX

Dedicated LCI progress threads:
- Better cache locality, cleaner worker thread behavior, more responsive and consistent network behavior.
- Instead of implicit background works in every MPI calls.

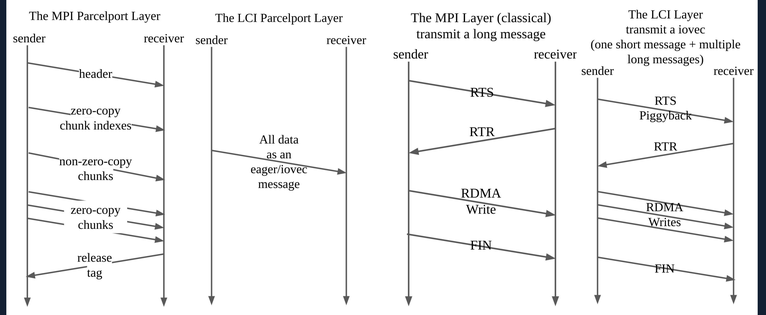
One-sided put operation:
- Only send. No receive needs to be posted.
- Receiver will get the references to the data from LCI **completion queues**.
- Instead of pre-posting multiple receives, probing, and then posting more follow-up receives in the MPI parcelport.

Directly put an "iovec" message:
- Send one eager message and multiple rendezvous messages in one call.
- Instead of multiple send/recv in the MPI parcelport.

Use Completion queues to deliver messages completion information:
- Instead of repeated MPI_Test for lots of MPI requests.

No mutex locks on HPX/LCI message passing path:
- Instead of multiple lock-protected layers in the HPX/MPI message passing path: parcel queues, connection cache, pending request store, and internal lock in MPI.

Other Optimizations:
- Thread-local backlog queues with aggregation.
- Separate network resources/progress threads for eager/rendezvous messages.



References

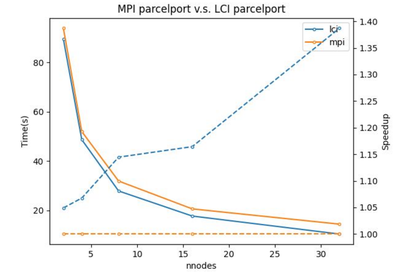
1. Dang, Hoang-Vu, Marc Snir, and William Gropp. "Towards millions of communicating threads." Proceedings of the 23rd European MPI Users' Group Meeting. 2016. (https://github.com/uiuc-hpc/LC/tree/dev-v1.7)
2. Kaiser, Hartmut, et al. "HPX: A task based programming model in a global address space." Proceedings of the 8th International Conference on Partitioned Global Address Space Programming Models. 2014. (https://github.com/STEllAR-GROUP/hpx)
3. Marcello, Dominic C., et al. "octo-tiger: a new, 3D hydrodynamic code for stellar mergers that uses hpx parallelization." Monthly Notices of the Royal Astronomical Society 504.4 (2021): 5345-5382. (https://github.com/STEllAR-GROUP/octotiger)

Evaluation

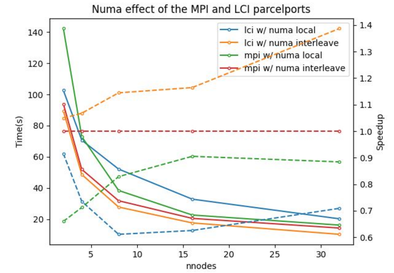
Platform: SDSC Expanse
- AMD EPYC 7742, 2 sockets, 4 NUMA nodes per socket, 128 cores (per node)
- 256GB Memory per node.
- HDR InfiniBand Interconnect.

Application: Octo-Tiger
- Astrophysics program simulating the evolution of star systems.
- Based on the fast multipole method on adaptive Octrees.

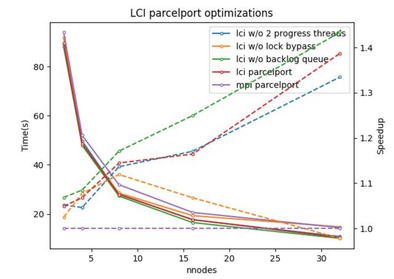
Results:
- LCI v.s. MPI:
  - The LCI parcelport surpassed the MPI parcelport by more than 40%.



- NUMA effects:
  - NUMA effects are significant.
  - LCI is more sensitive to NUMA effects than MPI.



- Specific optimizations:
  - Removing locks on the communication path helps. (~40% speedup)
  - Using separate resources helps. (~4% speedup)
  - The backlog queue doesn't
  - help. (~3% slowdown)



NSF

ILLINOIS

Efficient Message Passing Support for Asynchronous Many-Task Systems

Jiakun Yan[1], Hartmut Kaiser[2], Marc Snir[1]

[1]Department of Computer Science,, University of Illinois at Urbana-Champaign

[2]Center of Computation and Technology, Louisiana State University

Motivation

Modern Parallel Architecture:

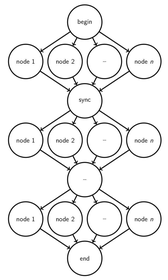
- Increased intra-node parallelism.
- Increased heterogeneity.
- Powerful Interconnect.

Task-based Programming Model:

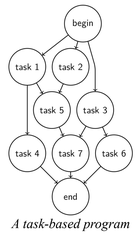
- Programmers decompose their programs into **tasks** along with their **dependencies**.
- The runtime will handle the **mapping**, **scheduling,** and **data movement**.
- E.g. HPX, Legion, PaRSEC.
- Communication layer: MPI/GASNet.

New Communication Pattern:

- Multithreaded.
- Irregular destinations.
- Small messages.



A BSP program



A task-based program

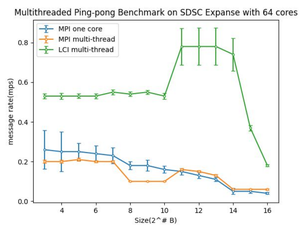
Lightweight Communication Interface (LCI) [1]

A low-level communication library.

- Intended user: high-level library developers.
- Like UCX/Libfabric/GASNet, as opposed to MPI.

Designed with task-based runtime as target clients.

- Applies to other irregular applications



Major features:

- **Flexible communication primitives and signaling mechanisms.**
- **Better multithreaded performance.**
- **Explicit user control of communication behaviors and**

HPX [2]

Task-based programming model with implicit task dependency graphs.

- Users invoke tasks like sequential code, and the runtime analyzes data usage and builds a task dependency graph.

Its communication layer implements a "parcelport" interface.

- A parcel consists of a small buffer (control data + small arguments) and optionally a few large zero-copy buffers.
- Current implementations: TCP, MPI, libfabric (work in progress).

The LCI Parcelport for HPX

Dedicated LCI progress threads:

- Better cache locality, cleaner worker thread behavior, more responsive and consistent network behavior.
- Instead of implicit background works in every MPI calls.

One-sided put operation:

- Only send. No receive needs to be posted.
- Receiver will get the references to the data from LCI **completion queues**.
- Instead of pre-posting multiple receives, probing, and then posting more follow-up receives in the MPI parcelport.

Directly put an "iovec" message:

- Send one eager message and multiple rendezvous messages in one call.
- Instead of multiple send/recv in the MPI parcelport.

Use Completion queues to deliver messages completion information:

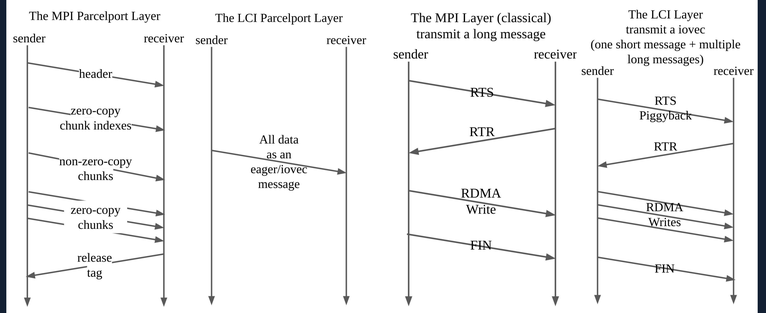
- Instead of repeated MPI_Test for lots of MPI requests.

No mutex locks on HPX/LCI message passing path:

- Instead of multiple lock-protected layers in the HPX/MPI message passing path: parcel queues, connection cache, pending request store, and internal lock in MPI.

Other Optimizations:

- Thread-local backlog queues with aggregation.
- Separate network resources/progress threads for eager/rendezvous messages.



References

1. Dang, Hoang-Vu, Marc Snir, and William Gropp. "Towards millions of communicating threads." Proceedings of the 23rd European MPI Users' Group Meeting. 2016. (https://github.com/uiuc-hpc/LC/tree/dev-v1.7)
2. Kaiser, Hartmut, et al. "HPX: A task based programming model in a global address space." Proceedings of the 8th International Conference on Partitioned Global Address Space Programming Models. 2014. (https://github.com/STEllAR-GROUP/hpx)
3. Marcello, Dominic C., et al. "octo-tiger: a new, 3D hydrodynamic code for stellar mergers that uses hpx parallelization." Monthly Notices of the Royal Astronomical Society 504.4 (2021): 5345-5382. (https://github.com/STEllAR-GROUP/octotiger)

Evaluation

Platform: SDSC Expanse

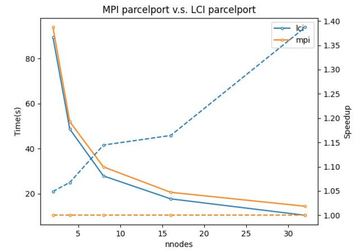
- AMD EPYC 7742, 2 sockets, 4 NUMA nodes per socket, 128 cores (per node)
- 256GB Memory per node.
- HDR InfiniBand Interconnect.

Application: Octo-Tiger

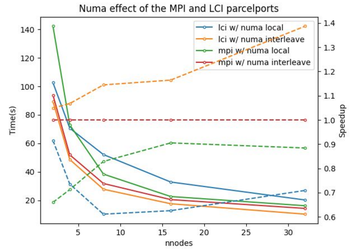
- Astrophysics program simulating the evolution of star systems.
- Based on the fast multipole method on adaptive Octrees.

Results:

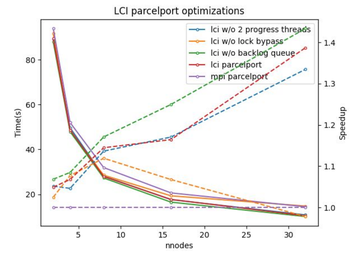
- LCI v.s. MPI:
  - The LCI parcelport surpassed the MPI parcelport by more than **40%**.



- NUMA effects:
  - NUMA effects are significant.
  - LCI is more sensitive to NUMA effects than MPI.



- Specific optimizations:
  - Removing locks on the communication path helps. (~40% speedup)
  - Using separate resources helps. (~4% speedup)
  - The backlog queue doesn't help. (~3% slowdown)



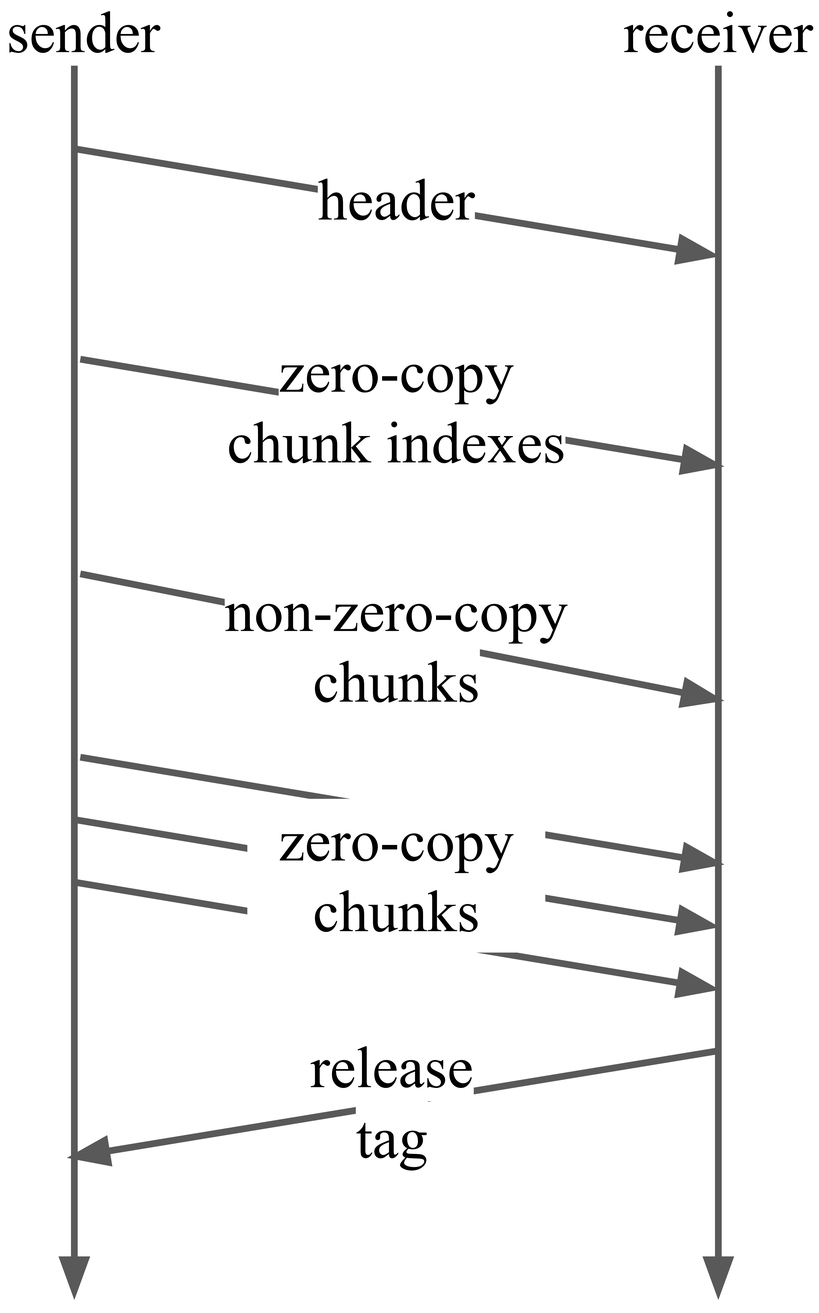
ACKNOWLEDGEMENTS

Check to make sure you've acknowledged partner and funding agencies, either with text or with their logos.

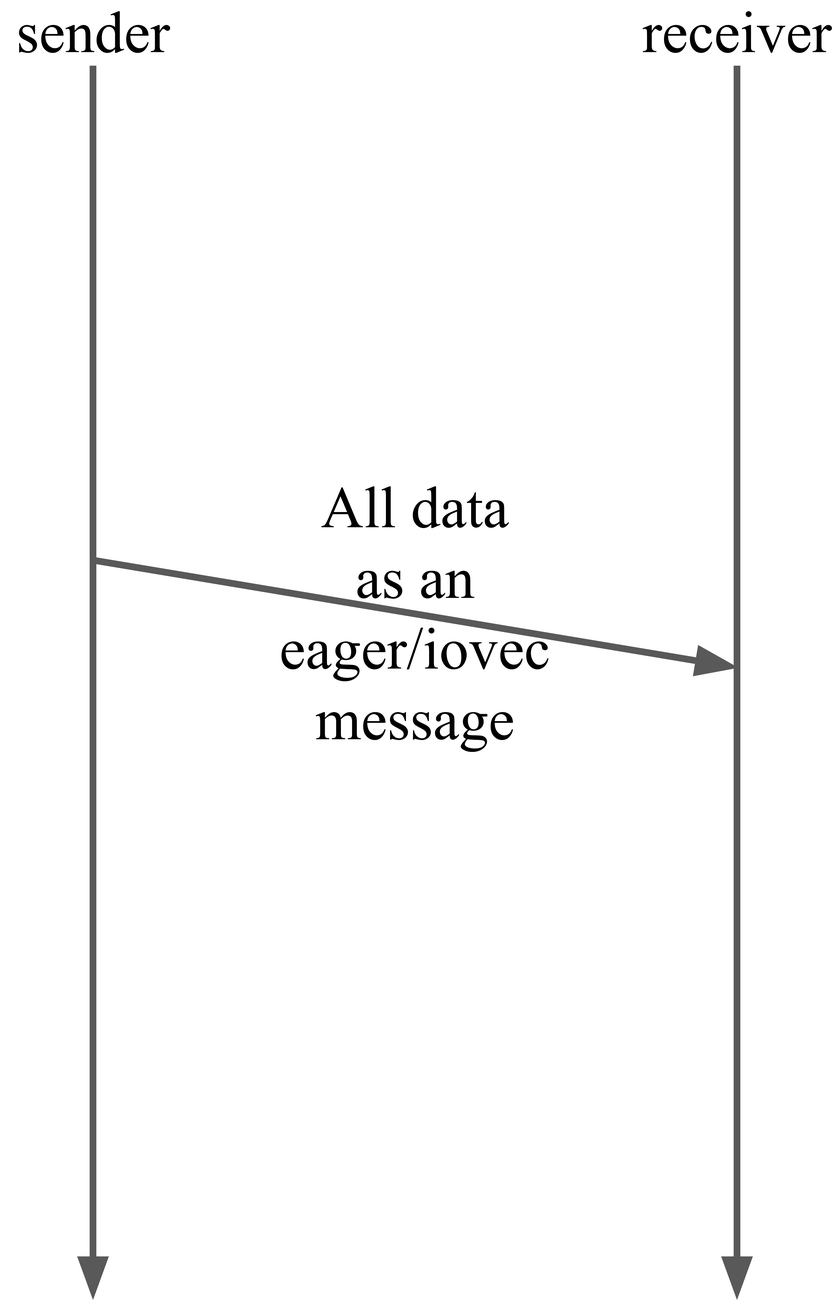
NSF

ILLINOIS

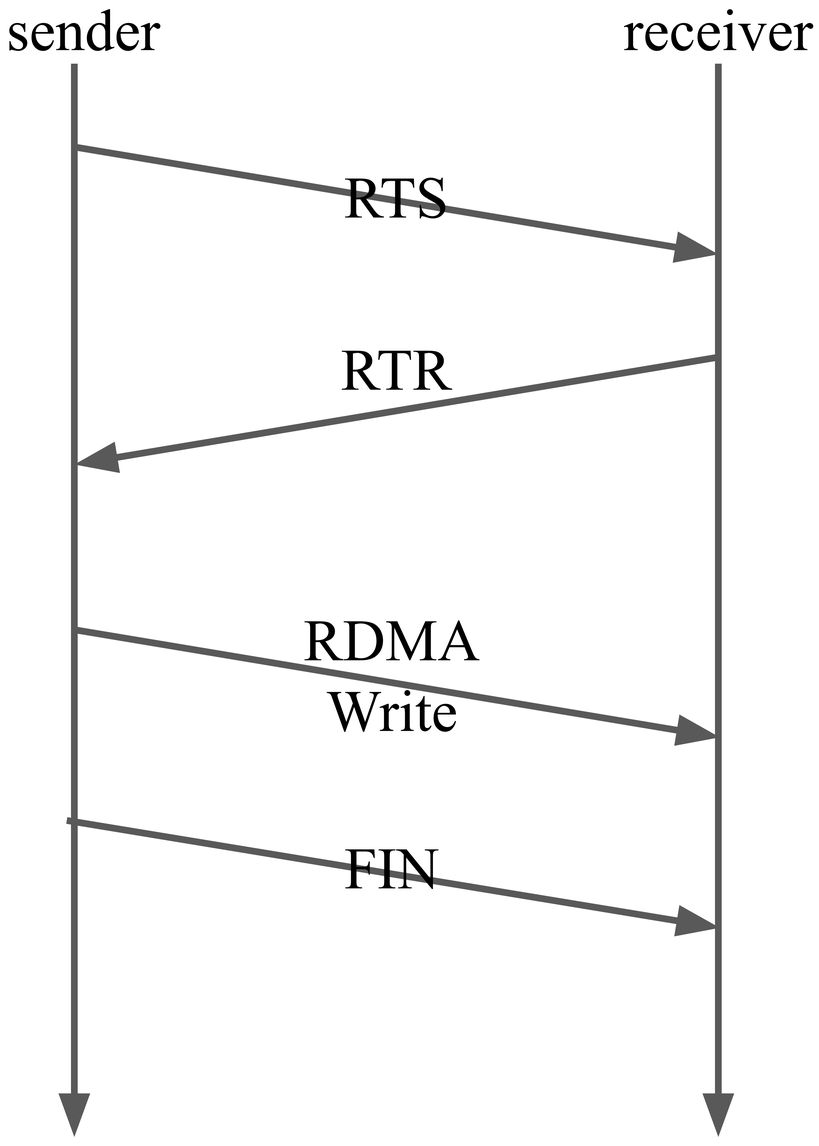
The MPI Parcelport Layer



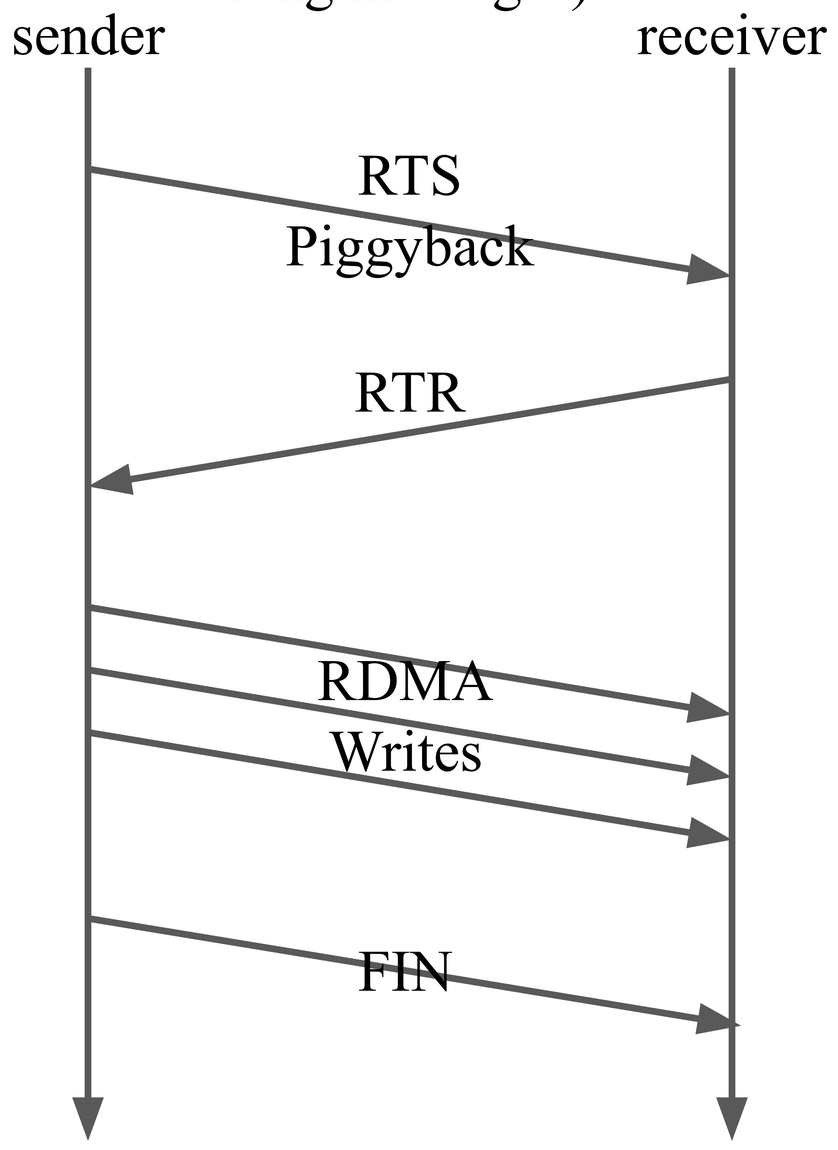
The LCI Parcelport Layer



The MPI Layer (classical)
transmit a long message



The LCI Layer
transmit a iovec
(one short message + multiple
long messages)



Commonly used logos