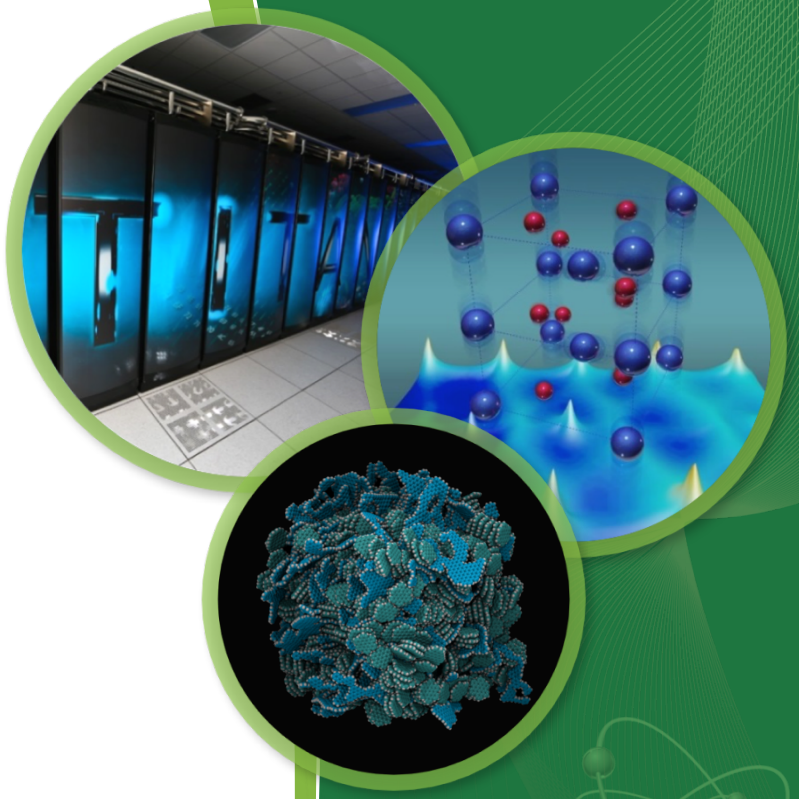
Merged Requests for Better Performance and Productivity in Multithreaded OpenSHMEM

Swen Boehm, Swaroop Pophale, Matthew B. Baker, and Manjunath Gorentla Venkata



August 7th, 2017

ORNL is managed by UT-Battelle
for the US Department of Energy

OAK RIDGE National Laboratory

Implicit Non-Blocking API

- OpenSHMEM currently supports implicit non-blocking API
- Requires completion by shmem_quiet
  - Finer granularity of synchronization not possible
- not enough abstractions to take advantage of modern HPC networks and multiple threads
  - Fine grained Progress for outstanding operations is not possible
  - Sharing resources is not possible

OAK RIDGE National Laboratory

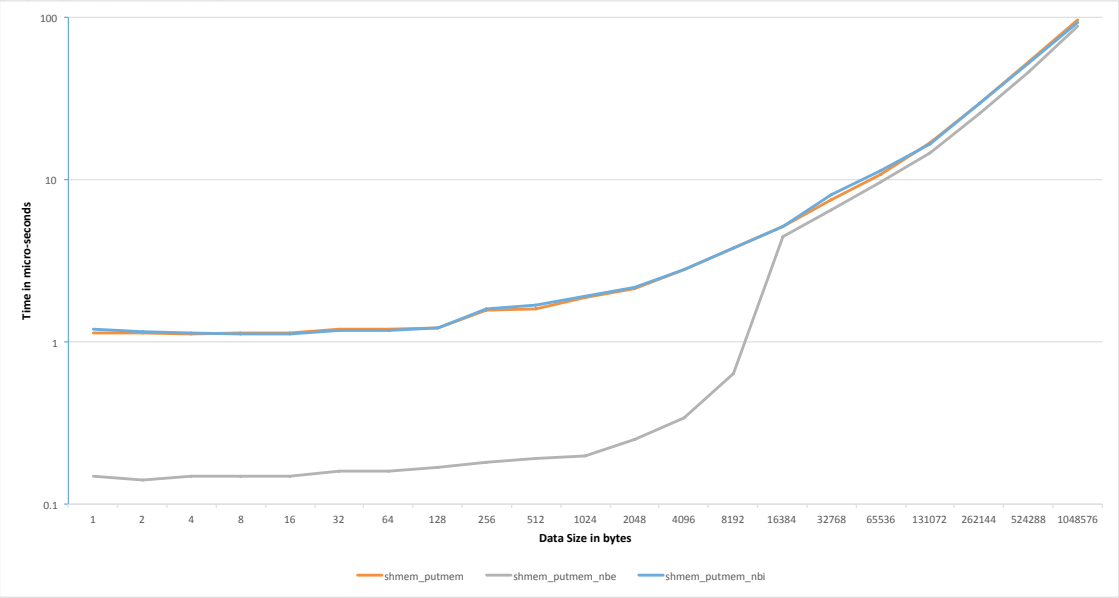
Motivation: Explicit Non-blocking API

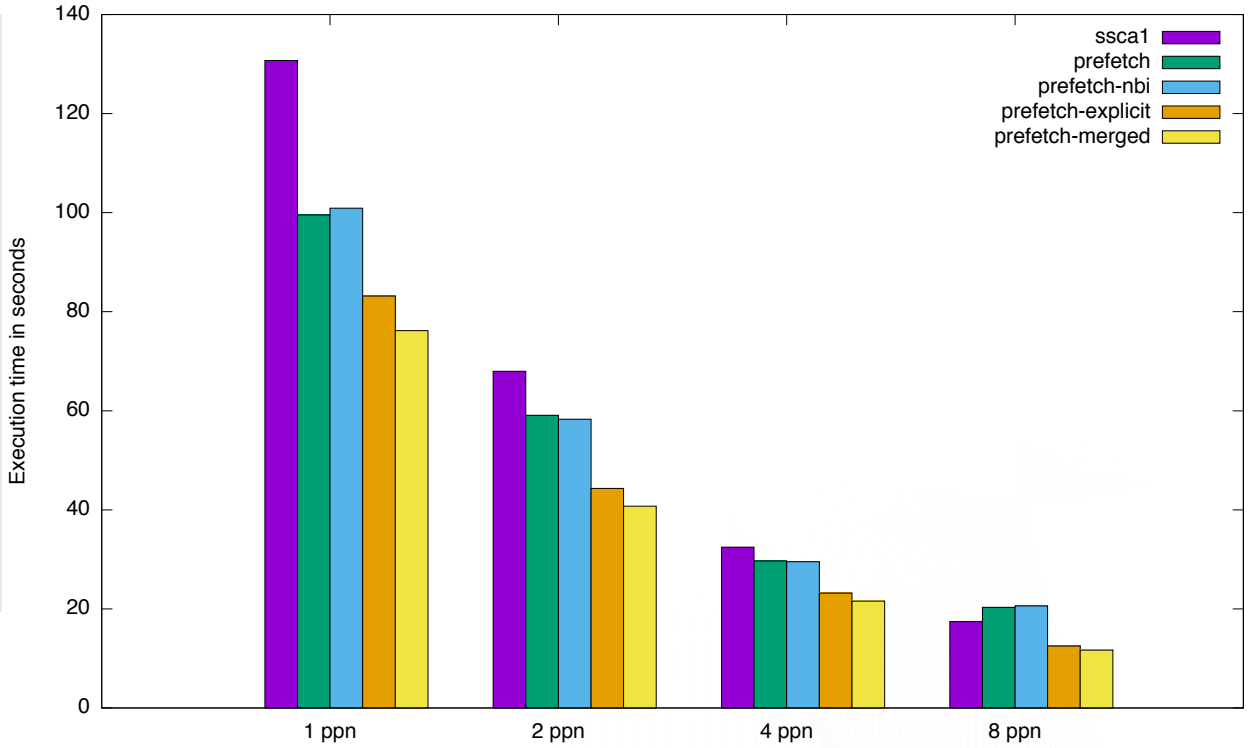
- Provide abstractions to take advantage of performance and parallelism of HPC networks
- Finer grained synchronization
- Ability to track individual / related operations

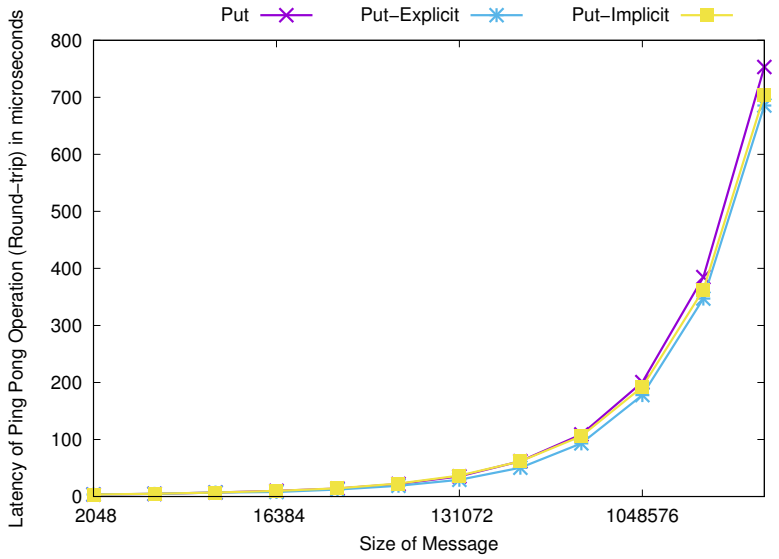
Criticism: explicit non-blocking RMA has overhead

OAK RIDGE National Laboratory

Explicit and implicit has similar performance







Associated overhead can be avoided with carful design

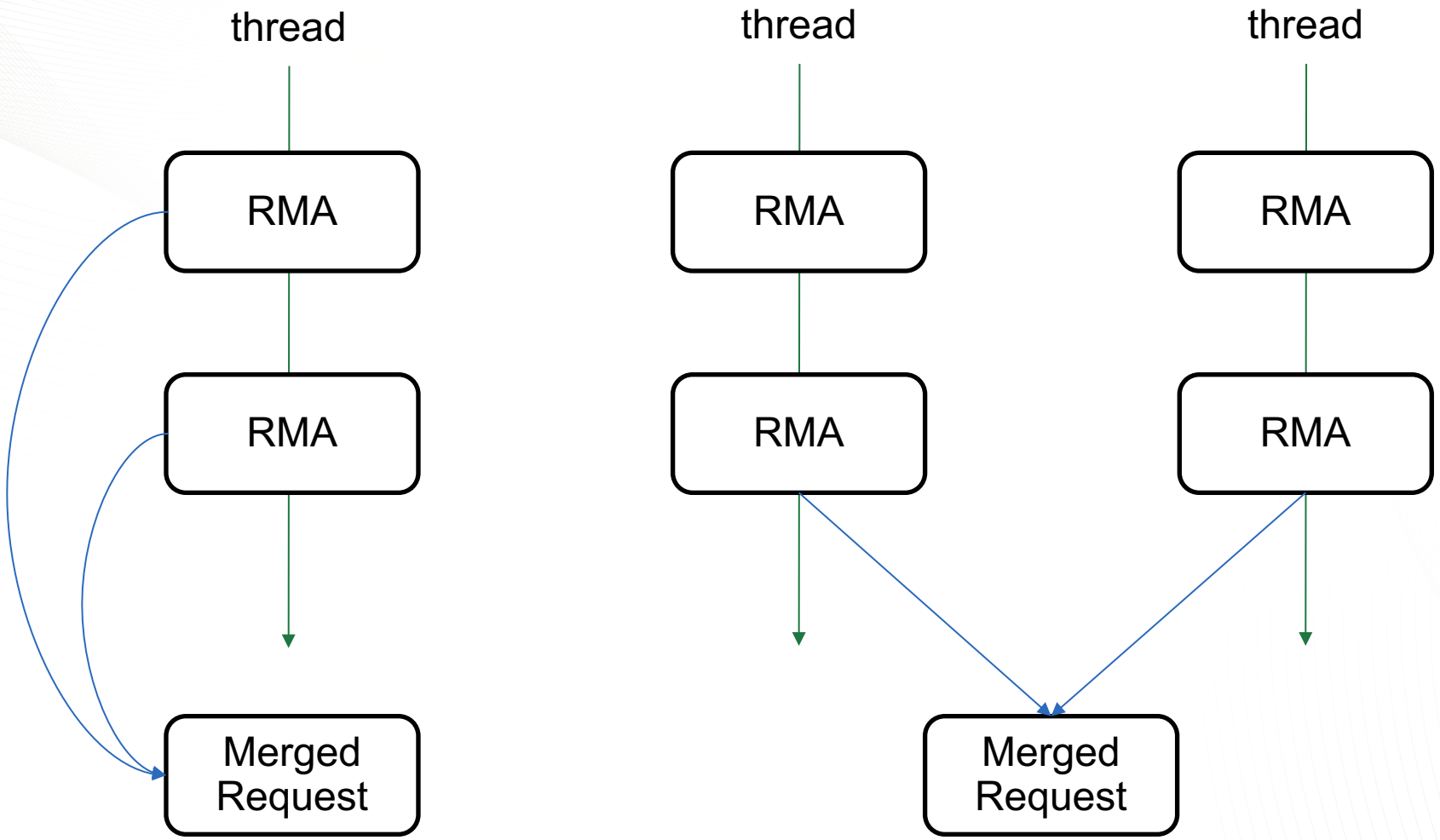
OAK RIDGE National Laboratory

Merged request

- Object associated a number of outstanding operations
- What does it enable the user to do
  - Streaming operations
  - Resource allocation
  - Threaded PEs uses

OAK RIDGE National Laboratory

Threaded PEs



OAK RIDGE National Laboratory

Explicit non-blocking RMA: API

- Single request handle for a collection of RMA operations

```
void shmem_TYPE_OP_nbe (TYPE *target, const TYPE *source, size_t nelems,
int pe, shmem_request_h *handle);
```

- Semantics
  - Create a request handle and can use the same handle for multiple operations
  - Merge a set of request handles
  - RMA operation is either a PUT or GET
  - Atomic operations (shmem_NAME_{swap,cswap,fadd}_nbe)
  - Completes with a call to wait

OAK RIDGE National Laboratory

RMA Operations – cont.

- Function to pre-allocate a request object
  - shmem_*_nbe will allocate a request object if NULL pointer is given for handle
  - Pre-allocating the handle lets user provide hints to the runtime and may enable optimizations

```
shmem_allocate_request (shmem_request_params_t  params,
                        shmem_request_h **handle);
```

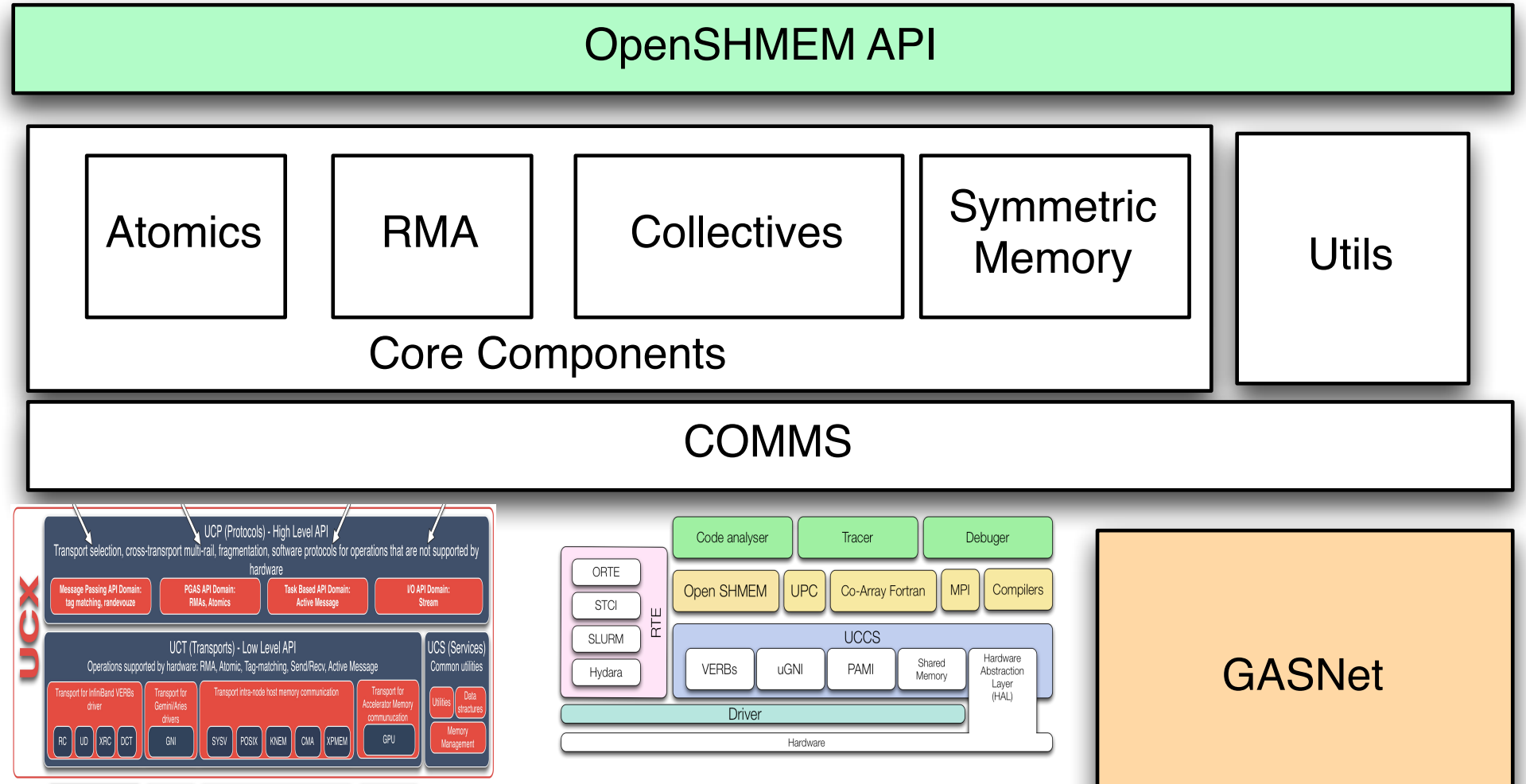
- Can be used to merge request handles into a merged handle
  - Possibility to isolate “slow” PEs
  - Can be used for Fault Tolerance

```
shmem_merge_request (size_t count, shmem_request_params_t *request,
                     void **handle);

shmem_free_request (shmem_request_h *handle);
```

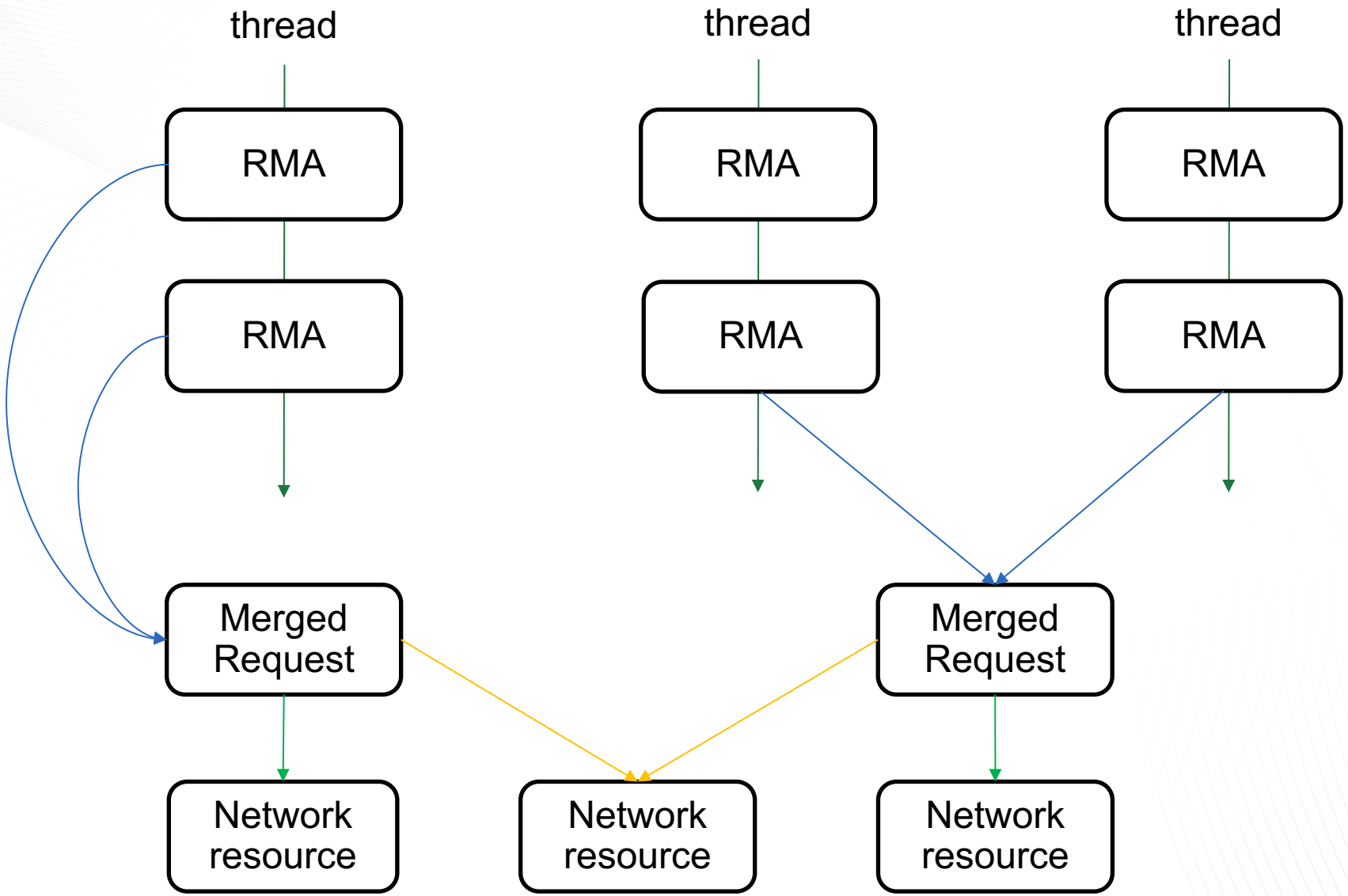

OAK RIDGE National Laboratory

Implementation



OAK RIDGE National Laboratory

Mapping of Request to Resources



OAK RIDGE National Laboratory

Experimental System: Turing

- 16 node cluster
- each node has
  - two IntelXeon E5-2660 processors
  - Mellanox ConnectX-4 VPI adapter card, EDR IB (100Gb/s)
  - 12GB RAM.
- Benchmarks
  - Modified OSU Message Rate micro-benchmark
  - Modified OSU Bandwidth micro-benchmark
  - User defined collective benchmark
  - Overlap benchmark
  - Modified Random Access benchmark

OAK RIDGE National Laboratory

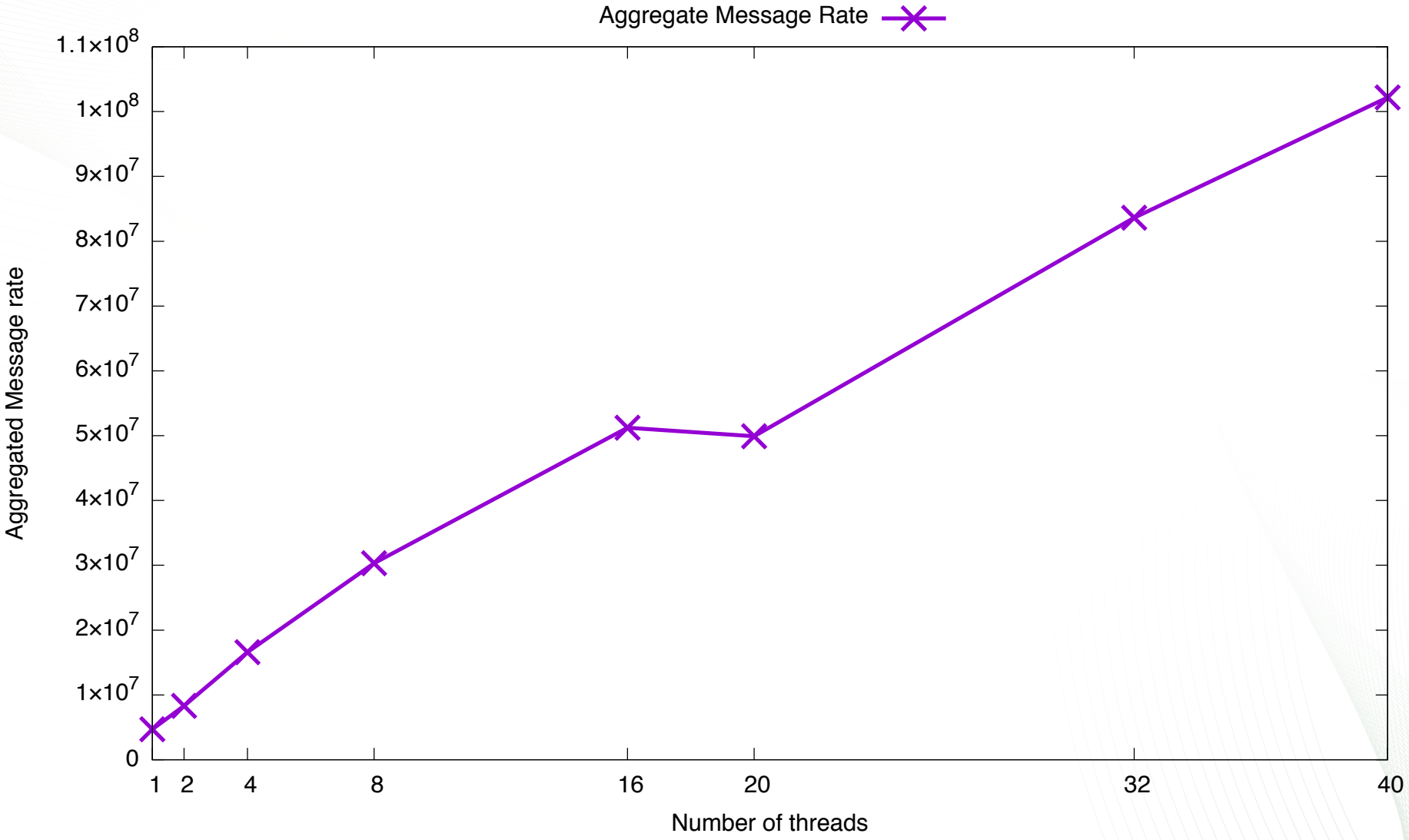
Evaluation: Non-blocking functions in multi threaded PEs

- Evaluate the ability to better utilize available resources
- Significant increase in message rate for small messages
- Message rate increased 20 fold using one PE running 40 threads compared to a single threaded PE on one node(100 MMPS vs. 4.5 MMPS)
- Twice the message rate if one PE w/ 20 threads is compared to 20 PEs on a node (50 MMPS vs 21 MMPS) – put this graph
- Can not reach the maximum supported by the HCA

MMPS = Million Messages Per Second

OAK RIDGE National Laboratory

Increasing Message rate (non-blocking puts)



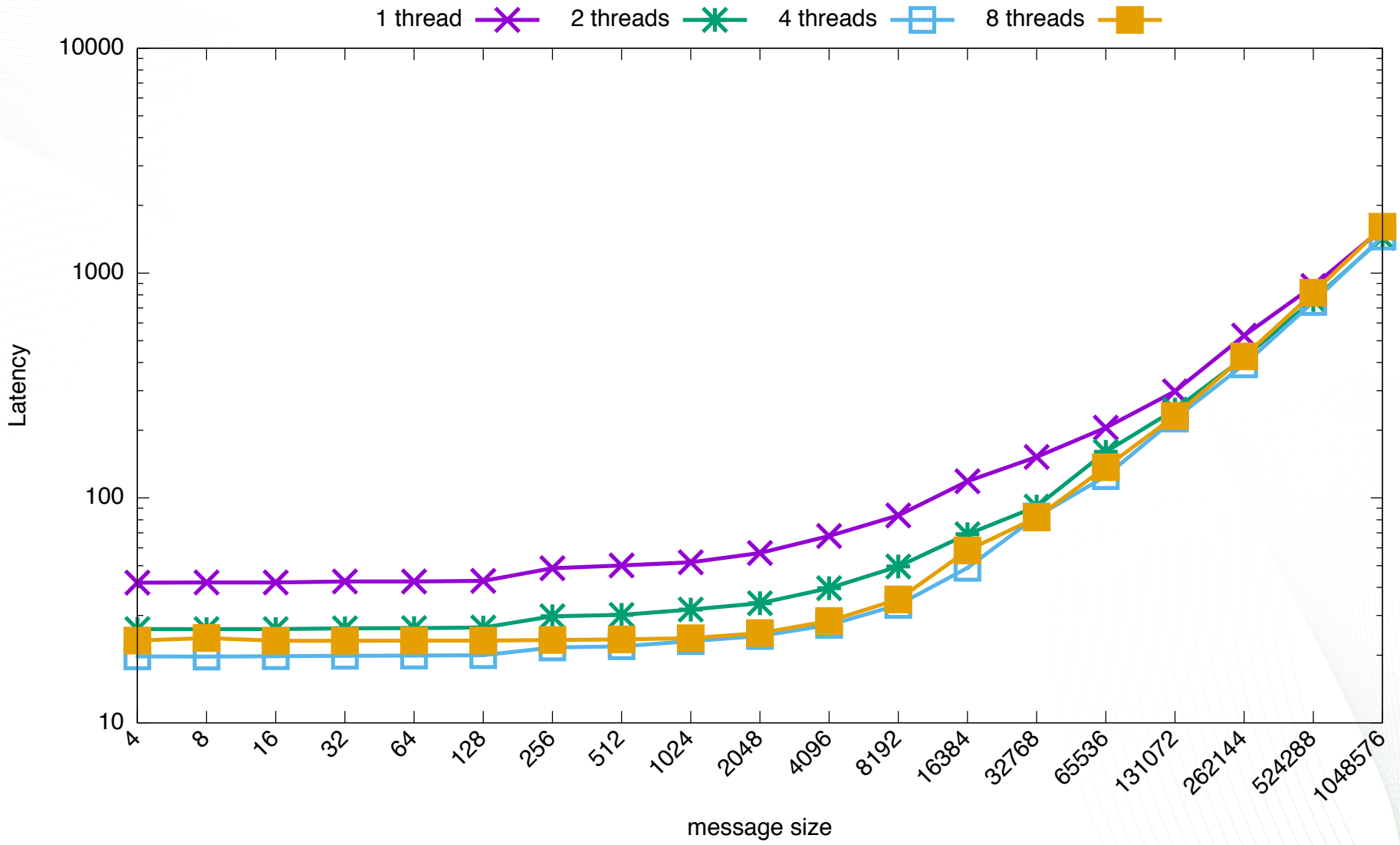
OAK RIDGE National Laboratory

Threaded collectives

- Implemented threaded broadcast algorithm on top of the API extensions
- Using threads to divide the work load can decrease the latency of broadcast operations

OAK RIDGE National Laboratory

User defined Collective



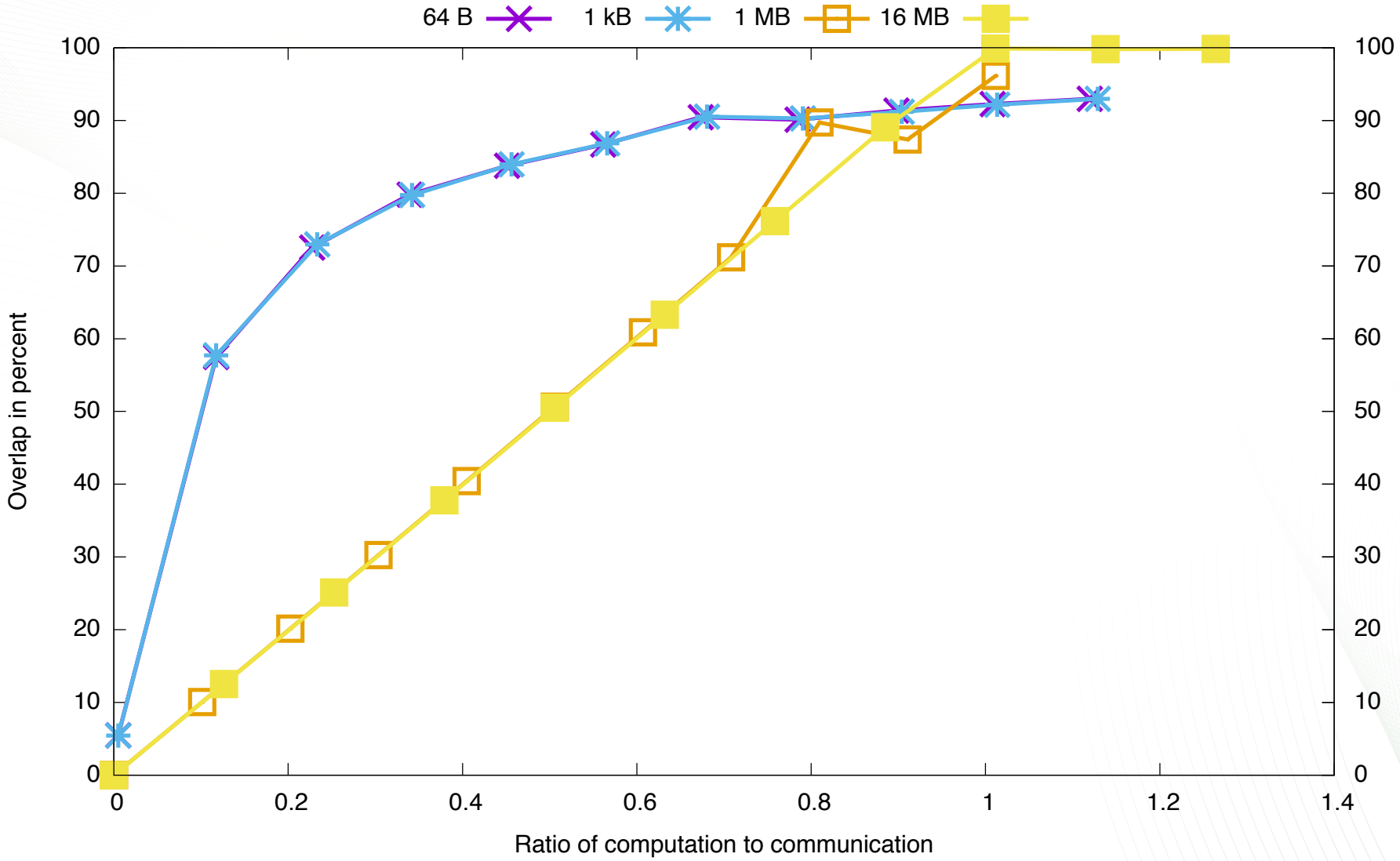
OAK RIDGE National Laboratory

Evaluation: Maximizing Overlap

- Why is it important
- Using non-blocking communication functions enables overlap of communication to computation
- We achieve almost 100% overlap for messages of 16 MB

OAK RIDGE National Laboratory

Overlap – non-blocking put



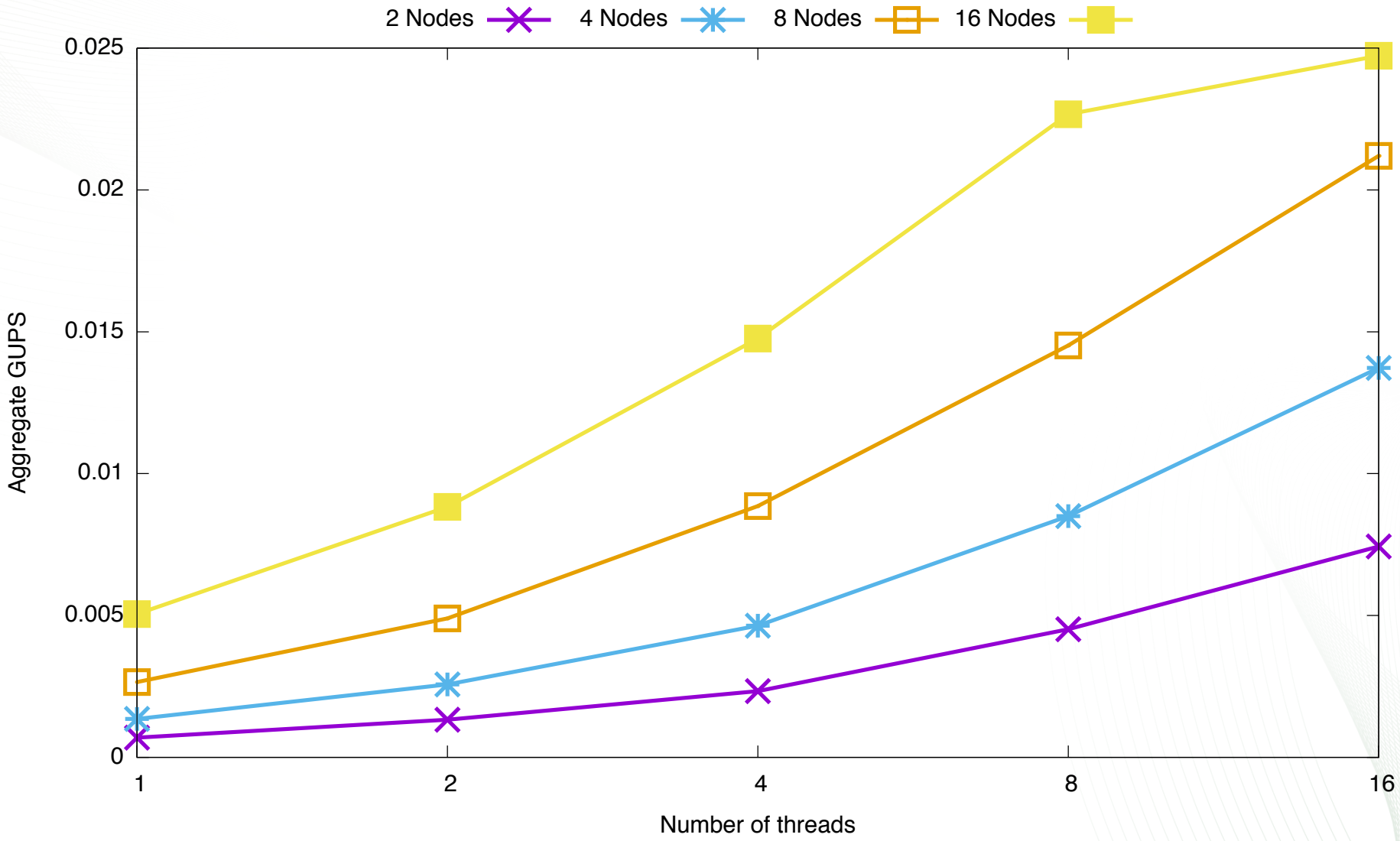
OAK RIDGE National Laboratory

Random Access

- Ported random access benchmark to support OpenSHMEM + OMP
- Random updates to the symmetric heap
- Symmetric heap set to half the nodes memory
- Outputs Giga Updates per second (GUPs)

OAK RIDGE National Laboratory

GUPS – non-blocking put/get



OAK RIDGE National Laboratory

Conclusion

- Using explicit RMA and merged requests with multiple threads
  - can achieve higher message rate
  - higher bandwidth for smaller message
  - better computation-communication overlap
- Used merged request to realize multithreaded collectives
  - multiple threads co-operate to complete the collective operation

OAK RIDGE National Laboratory

Questions?

OAK RIDGE National Laboratory