GASNet-EX Performance Improvements Due to Specialization for the Cray Aries Network

Paul H. Hargrove and Dan Bonachea

gasnet-staff@lbl.gov https://gasnet.lbl.gov

Parallel Applications Workshop, Alternatives to MPI (PAW-ATM)
November 16, 2018

Abstract

- Introduce GASNet-EX, the successor to GASNet-1
- Show performance improvements due to implementation specific to the Cray Aries network

GASNet-1: Overview

- Started in 2002 to provide a portable network communication runtime for three PGAS languages:
  - UPC, CAF and Titanium
- Primary features:
  - Non-blocking RMA (one-sided Put and Get)
  - Active Messages (simplification of Berkeley AM-2)
- Motivated by semantic issues in (then current) MPI-2.0
  - Dan Bonachea, Jason Duell, "Problems with using MPI 1.1 and 2.0 as compilation targets for parallel language implementations", IJHPCN 2004.

GASNet: Adoption and Portability

- Client runtimes

LBNL UPC++	Stanford Legion	OpenSHMEM reference impl.
Berkley UPC	Titanium	Omni XcalableMP
GCC/UPC	Rice Co-Array Fortran	At least 7 others cited in the paper
Clang UPC	OpenUH Co-Array Fortran	
Cray Chapel	OpenCoarrays in GCC Fortran	

- Network conduits

OpenFabrics Verbs (InfiniBand)	IBM PAMI (BG/Q and others)	SHMEM (Cray X1 and SGI Altix)
Mellanox MXM and VAPI (InfiniBand)	IBM DCMF (BG/P)	Quadric elan3/4 (QsNet I/II)
Cray uGNI (Gemini and Aries)	IBM LAPI (Colony and Federation)	Myricom GM (Myrinet)
Intel PSM2 (OmniPath)	Cray Portals3 (Seastar)	Dolphin SISCI
UDP (any TCP/IP network)	OFI/libfabric	Shared memory (no network)
MPI 1.1 or newer	Sandia Portals4	

- Supported platforms
  - Over 10 compiler families, 15 operating systems and dozens of architectures

* These lists and counts include both current and past support

GASNet-EX: Overview

- GASNet-EX is the next generation of GASNet
  - Addressing needs of newer programming models such as LBNL UPC++, Stanford Legion and Cray Chapel
  - Incorporating over 15 years of lessons learned
  - Provides backward compatibility for GASNet-1 clients
- Motivating goals include
  - Support more client asynchrony
  - Enable more client adaptation
  - Improve memory footprint
  - Improve threading support
  - Support offload to network h/w
  - Support multi-client applications
  - Support for device memory

GASNet-EX: Status

- GASNet-EX is a work-in-progress
  - Not every new feature has been implemented yet
  - Many have, with benefits this presentation will show
- Three key clients using GASNet-EX
  - UPC++ v1.0 requires GASNet-EX
  - Legion and Chapel are starting work to use EX features
- Will displace legacy GASNet-1 implementation in 2019

Specific GASNet-EX Improvements

- This talk reports on four of GASNet-EX's new features
  - Local Completion Control
  - Immediate-mode Communication Injection
  - Negotiated-payload Active Messages
  - Remote Atomics
- GASNet-EX includes a network-independent "reference implementation" of each of these
- Here we show the performance on Cray XC40 systems due to "specialization" in GASNet-EX's "aries-conduit"
- The paper provides more detail than can be given here

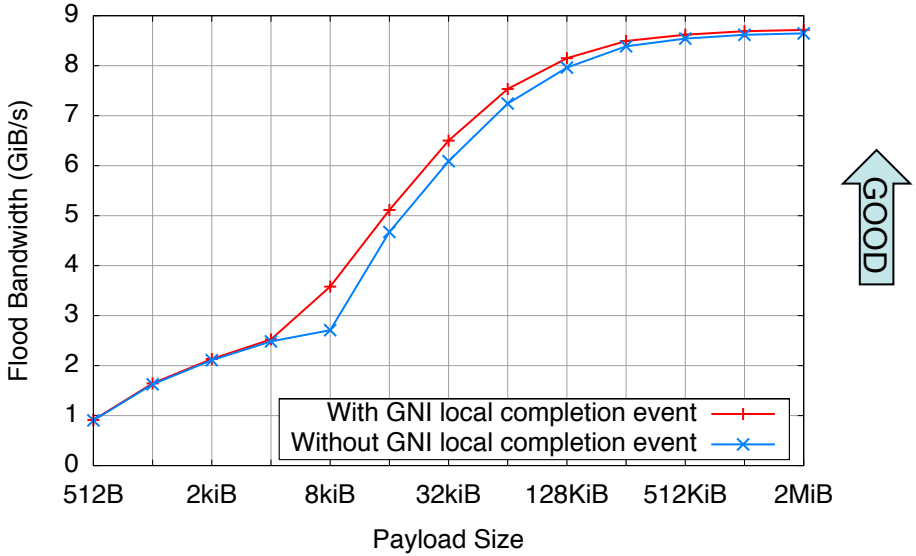
Local Completion Control

- GASNet-EX introduces means for client to test (or wait) for local completion *between* injection and completion of a non-blocking Put
  - Here “local completion” refers to the point in time when it is safe to overwrite the source of a Put or Active Message
  - This may occur independent of operation completion
- Exposes greater opportunity for communication overlap than possible with the “bulk” and “non-bulk” options available in GASNet-1

Local Completion Control

- Figure shows a proxy for how exposing a local completion event can improve overlap:
  - The analogous change *within* GASNet-EX's aries-conduit has improved flood bandwidth
- Blue series shows bandwidth prior to utilizing GNI-level local completion
- Red series shows up to 32% increased bandwidth with the local completion event

Non-bulk Put flood bandwidth on Cray Aries with and without use of a local completion event at the GNI level



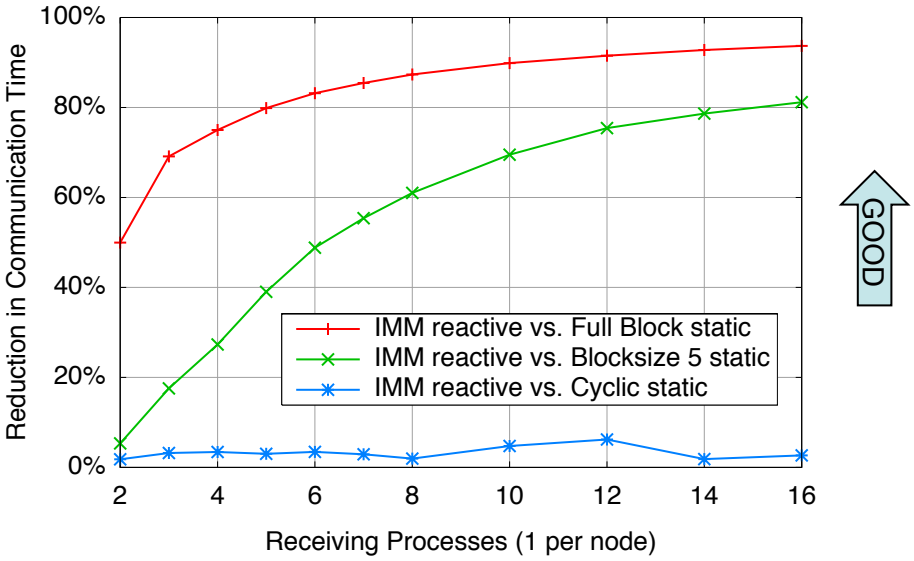
Immediate-mode Communication Injection

- Lack of resources can stall communication injection
  - Such backpressure may be path-specific
- New feature allows client adaptation to such a scenario
  - E.g. work-stealing could select a different victim
- Immediate-mode is a flag which permits (does not require) implementation to return *without* performing communication, in the presence of backpressure

Immediate-mode Communication Injection

- Figure illustrates performance on a benchmark modeling AM communication with inattentive peers
- Shows reduction in time to complete communication using a “reactive” immediate-mode approach
- The series compare reactive to three distinct static schedules
- Best case is 93% reduction

Reduced communication delays using immediate-mode Active Messages



Negotiated-Payload Active Messages

- "Negotiated-Payload" is a new family of AM interfaces
  - Splits AM injection into distinct Prepare and Commit phases
  - Client and GASNet can negotiate the buffer size and ownership
- Case 1: "chunking" loops may better utilize available buffer resources, allowing fewer larger messages
- Case 2: remove critical-path **`memcpy`** for some patterns

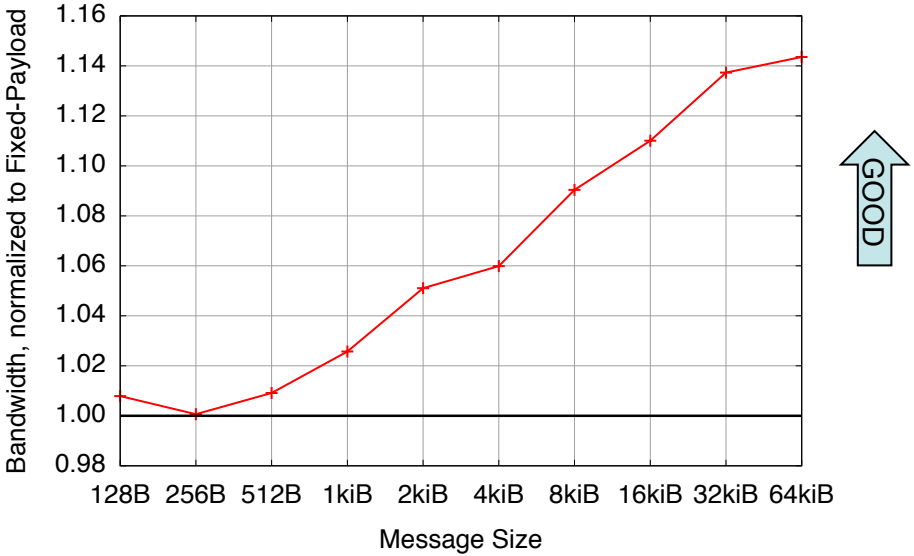
```
// Fixed-Payload code, for which most conduits require a memcpy to an internal buffer:
assemble_payload(client_buf, len);  // writes client-owned memory
gex_AM_RequestMedium1(team, rank, handler, client_buf, len, GEX_EVENT_NOW, flags, arg);

// Negotiated-Payload avoids the memcpy via payload assembly into a GASNet-owned buffer:
gex_AM_SrcDesc_t sd = gex_AM_PrepareRequestMedium(team, rank, NULL, len, len, NULL, flags, 1);
assemble_payload(gex_AM_SrcDescAddr(sd), len);  // writes GASNet-owned memory
gex_AM_CommitRequestMedium1(sd, handler, len, arg);
```

Negotiated-Payload Active Messages

- Figure shows an AM ping-pong bandwidth benchmark using the memcpy-removal pattern on the previous slide
- Normalized to the Fixed-Payload performance
- Shows NP-AM implementation for Cray Aries network delivering up to a 14% improvement

Aries-conduit NP-AM speedup on a ping-pong test with dynamically-generated payload



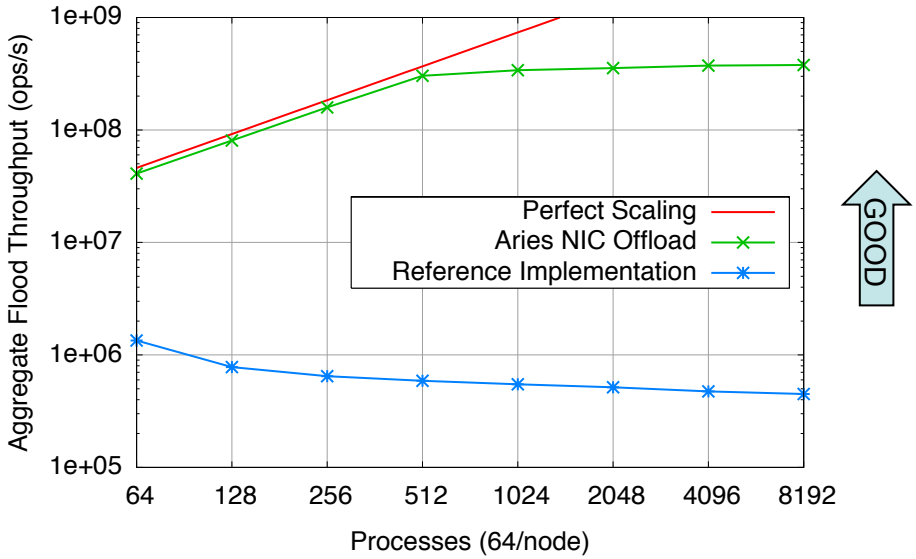
Remote Atomics

- "Remote Atomics" is a new family of RMA interfaces
  - Analogous to MPI accumulate operations
- Interface designed with offload in mind
- Uses the "atomics domain" concept
  - Introduced by UPC 1.3
  - Enables efficient offload, even in the presence of concurrent updates to the same location using multiple distinct operations

Remote Atomics

- Offload reduces latency of fetch-and-add by **70%** relative to generic AM-based reference
- Figure shows aggregate throughput of a “hot-spot” test of fetch-and-add (all to one)
- Green series shows robust scaling to saturation when offloaded to the Aries NIC

Scaling of a remote atomics
“hot-spot” test in the Cray Aries network



Conclusions

- GASNet-EX is the next generation of GASNet, addressing needs of newer programming models
  - Asynchrony, adaptively, threading, scalability, device memory, ...
- Already in production use by UPC++
  - Looking for new clients, talk to me over lunch!
- Provides backward compatibility for GASNet-1 clients
- Benefits of new features are already measurable

THANK YOU

gasnet-staff@lbl.gov

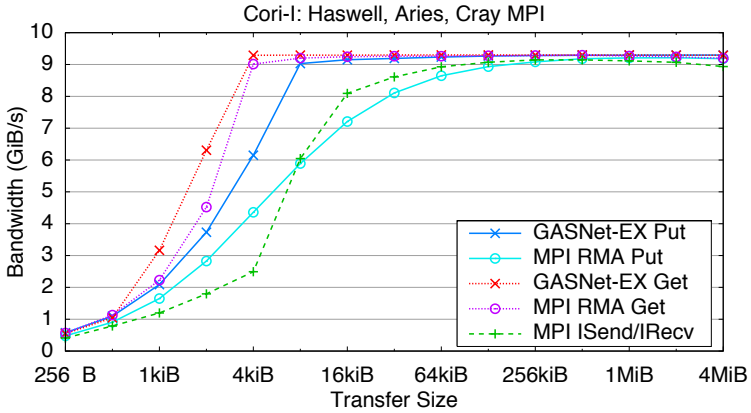
https://gasnet.lbl.gov

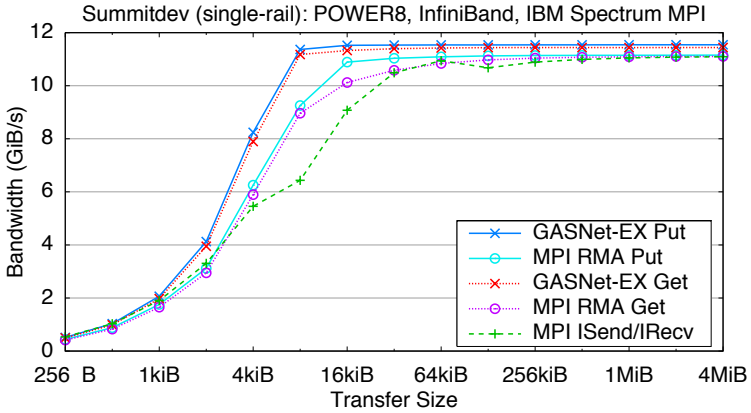
Acknowledgements

- This research was funded in part by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration.
- This research used resources of the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.
- This research used resources of the Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC02-06CH11357.

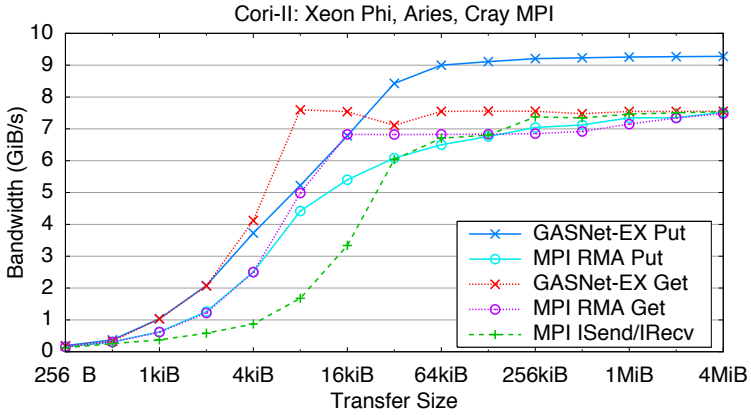
BACKUP SLIDES

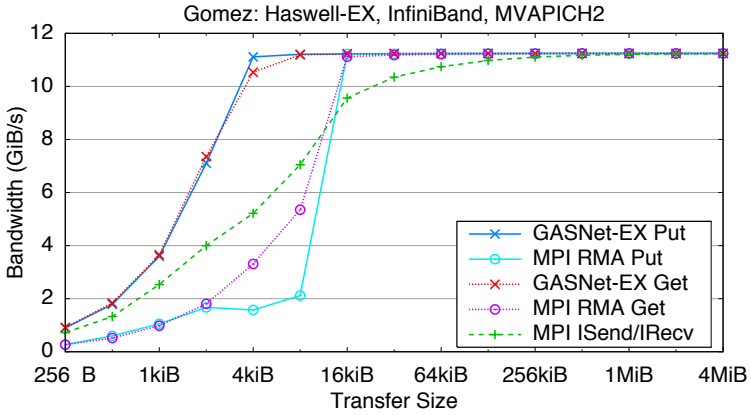
RMA Bandwidth Microbenchmarks





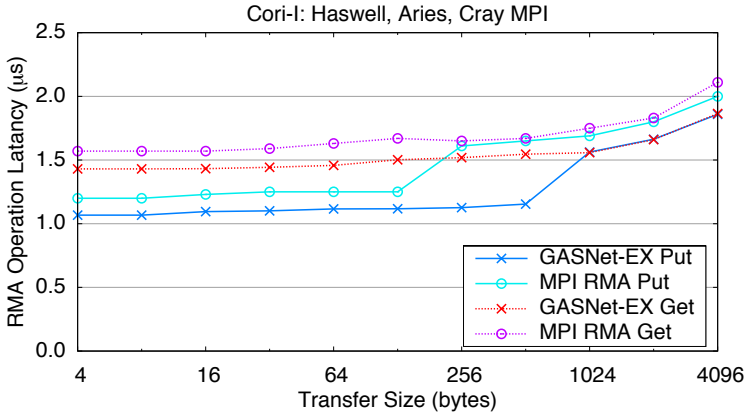
GOOD

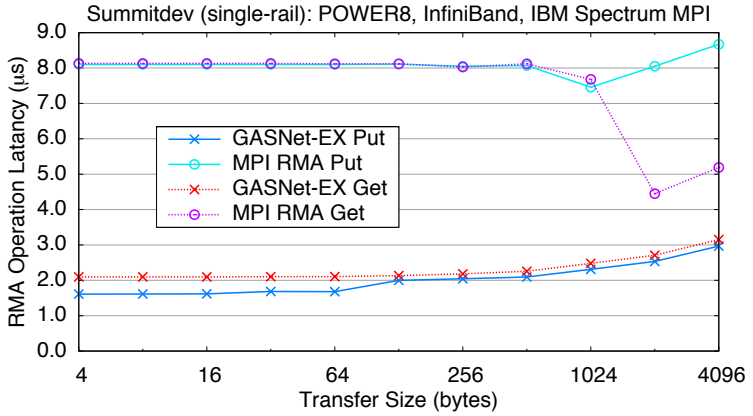




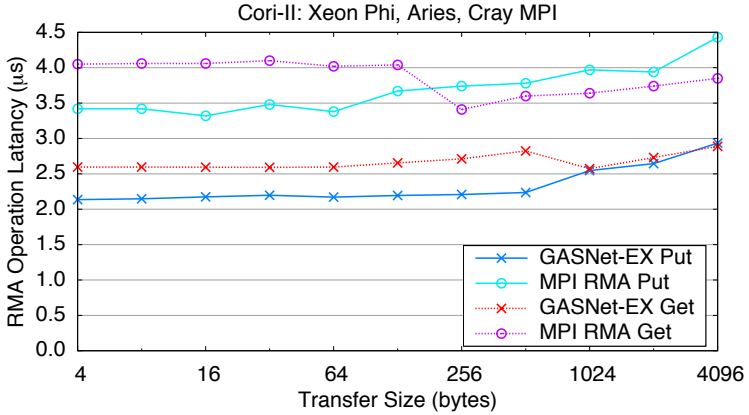
BERKELEY LAB

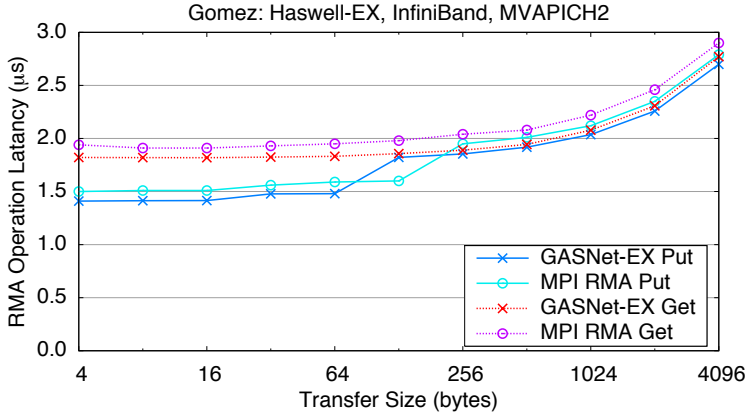
RMA Latency Microbenchmarks





GOOD





BERKELEY LAB