PGAS Programming Models: My 20-year Perspective

Paul H. Hargrove

https://go.lbl.gov/paul-hargrove

Outline

I. My early career or “why I drank the PGAS Kool-Aid”

II. The PGAS community 2001 to 2017

III. 2017 to the present

IV. Closing

Early Career 1

- Summer 1999 I had no funding from my Ph.D. adviser
- Was working in computational solid state physics
- Met Bill Saphire of LBNL† through a job fair
- Bill needed somebody to work on Linux networking, including kernel drivers
- I had good background in operating systems and hobbyist experience in the Linux kernel
- Summer position became career track a year later

† LBNL: Lawrence Berkeley National Laboratory. Will mostly be read as “Berkeley Lab” from here forward.

Early Career 2

- Worked on “M-VIA”
  - Modular Virtual Interface Architecture
  - VIA was a precursor to today’s InfiniBand standard
- Most details not terribly relevant, other than two:
  - RDMA: the ability of network hardware to move data in or out of address spaces of a user process without CPU participation in the critical path (also known as “zero copy”)
  - OS Bypass: the ability for network hardware to accept work from a user process without going through a system call

Early Career 3

- MVICH was a LBNL port of MPICH over VIPL
  - Work by Mike Welcome under the same funding
- As M-VIA funding came to a close, proposed LDRD† to port MVICH to InfiniBand
  - Was declined
  - Was told vendors would do the work if it was really needed
  - Ironically, D.K. Panda at OSU did the work instead: MVAPICH
- Instead joined Kathy Yelick's UPC project at LBNL
  - Ported GASNet to InfiniBand

†LDRD: Laboratory Directed R&D. Research funding managed at the Lab level, funded out of overhead

Early Career 4

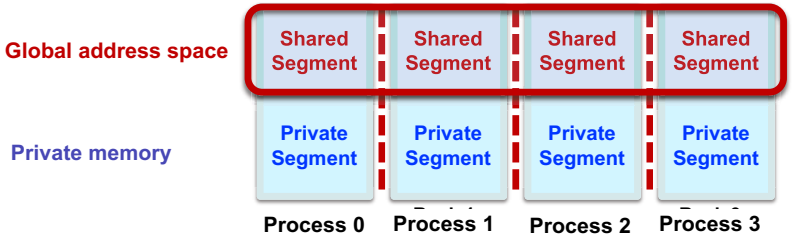
- PGAS was already an active area by this time with three languages having emerged in the late 1990s
  - UPC, C based
  - Co-array Fortran (CAF), Fortran based
  - Titanium (Ti), Java based

- GASNet began as a common network runtime for these
  - Topic of my talk at CHIUW'22 last year
- But what is "PGAS"?

The PGAS model

Partitioned **G**lobal **A**ddress **S**pace

- Support global memory abstraction
  - leveraging the network's RDMA capability
- Often distinguish private and shared memory
- Separate synchronization from data movement



Languages that provide PGAS:

Chapel, UPC, Fortran coarrays (Fortran 2008+), X10, Titanium…

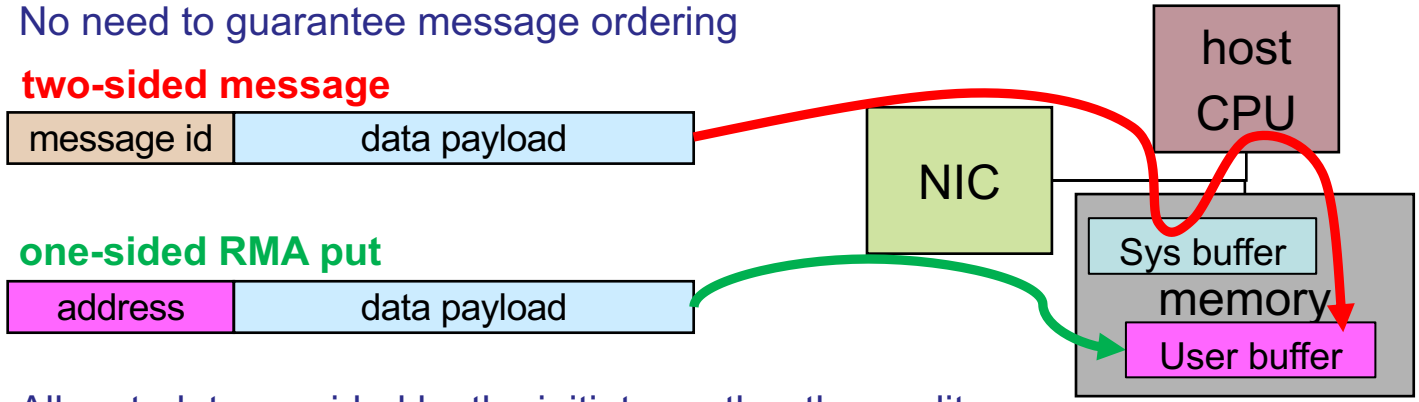
Libraries that provide PGAS:

UPC++, OpenSHMEM, Co-Array C++, Global Arrays, DASH…

A key semantic property is support for one-sided RMA

Reducing communication overhead using one-sided RMA

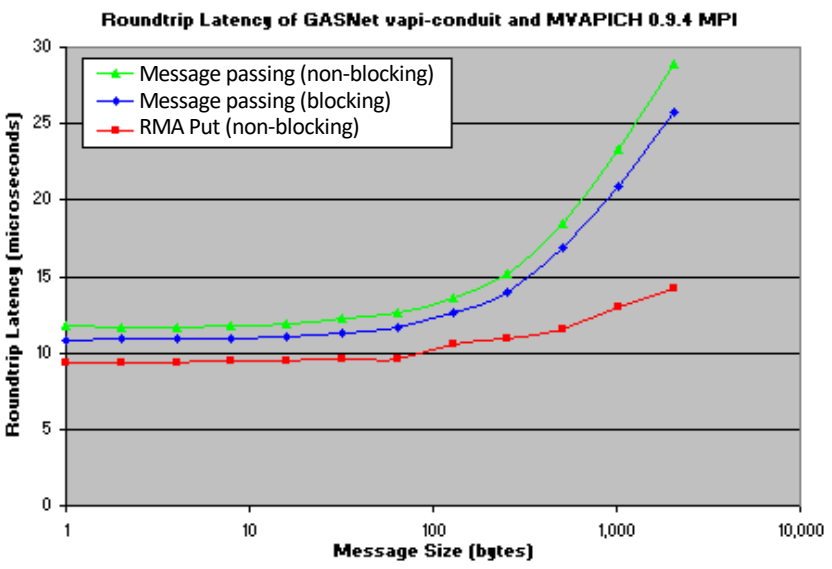
- Idea: Let each process directly access another's memory via a global pointer
- Communication is **one-sided :** there is no "receive" operation
  - No need to match sends to receives
  - No unexpected messages
  - No need to guarantee message ordering

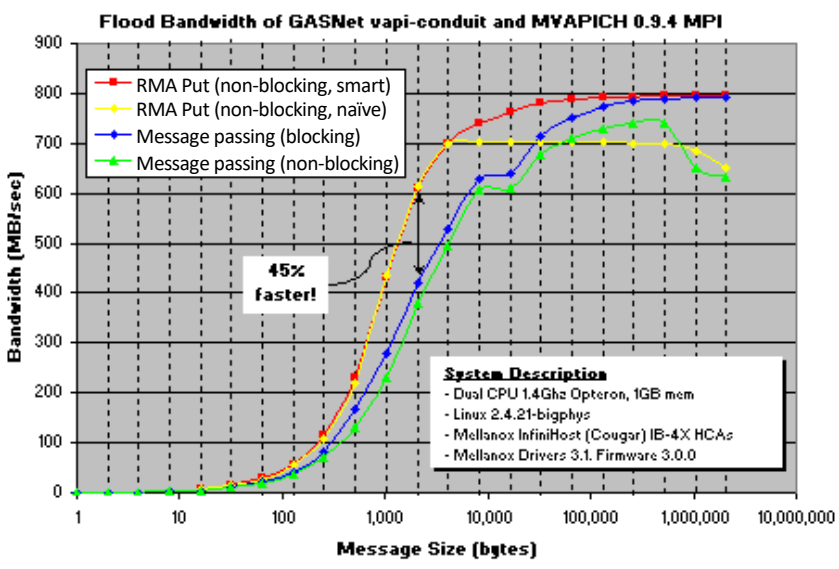


  - All metadata provided by the initiator, rather than split between sender and receiver
  - Supported in hardware through RDMA (Remote Direct Memory Access)
- Looks like shared memory: shared data structures with asynchronous access

November 2004 GASNet vs MVAPICH

Our oldest RMA vs message passing comparison I could find:





Among the things that helped me feel I'd "found my calling"

https://gasnet.lbl.gov/performance-2004/#lambda

Outline

I. My early career or “why I drank the PGAS Kool-Aid”

II. The PGAS community 2001 to 2017

III. 2017 to the present

IV. Closing

SCxy (Supercomputing) Booth

- UPC and PGAS booths
  - Began as SC01 UPC booth run by Tarek El-Ghazawi of GWU
  - I became involved in booth operations starting at SC05
  - Name changed from “UPC” to “PGAS” at SC07
  - LBNL took over operation from SC10 to SC17
- Booth features
  - Poster gallery highlighting work in the field
  - Flyers and CD/DVD in the early years
  - Gathering place for folks working in the field
- UPC and PGAS BOFs at SC as well

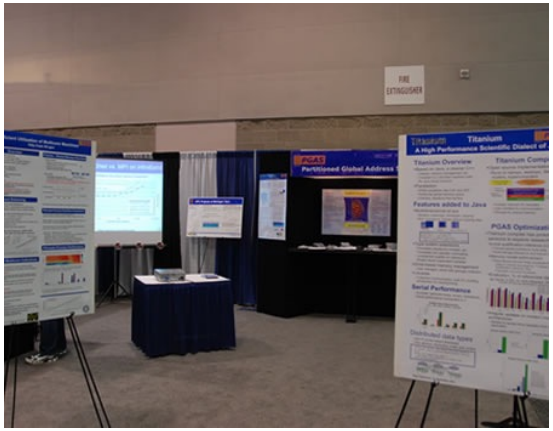


Image credit: George Washington University https://hpcl.gwu.edu/2017/10/11/pgas-research-booth-at-sc09/

Burton Smith and PGAS

- Co-founded Tera Computer Company in 1987
- In 2000, Tera bought Cray Research business unit from SGI to form Cray Inc.
- P.I. on Cray's DARPA HPCS project "Cascade"
- Left Cray for Microsoft in 2005
- Became an annual visitor to UPC/PGAS booth at SC

Image credit:
By Dimitrij Krepis - Taken by Dimitrij Krepis at Supercomputing 2007, CC BY-SA 2.5, https://commons.wikimedia.org/w/index.php?curid=3152007

DARPA HPCS Project

- High *Productivity* Computing Systems
- 2002 through 2012
- Goals included:
  - Multi-petaflops system(s)
  - 10x improvement in user productivity
  - Recognizes that “time to solution” involves both factors
- Three PGAS programming languages emerged
  - Chapel, X10, and Fortress
  - Chapel remains the most active today

Selected Chapel History

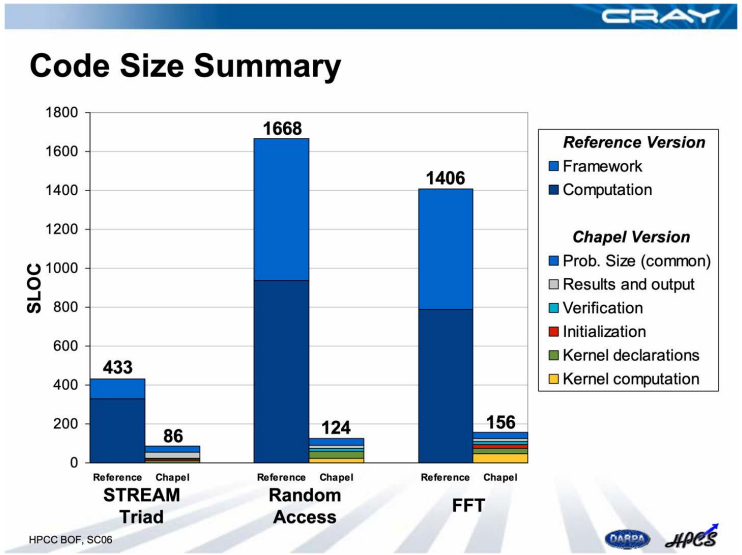
- April 2004
  - "The Cascade High Productivity Language" at HIPS04 https://doi.org/10.1109/HIPS.2004.1299190
  - Random observation:
    - The immediately preceding paper in those proceedings is on ZPL, co-authored by Brad Chamberlain and its Related Work section reviews UPC, CAF & Ti
- December 2006
  - Chapel 0.4 released with only single-locale support
- March 2008
  - Chapel 0.7 released with multi-locale support based on GASNet 1.10.0
- GASNet remains Chapel's recommended backend for InfiniBand

HPC Challenge (HPCC) Awards

- Product of the HPCS Project
  - Ran 2005 through 2014
- Application kernels and communications benchmarks
- In spirit of HPCS goals, annual award in two classes
  - Class 1: Best Performance
  - Class 2: Most Productive (a subjective “beauty contest”)
- Class 1 winners mostly non-portable hand-tuned codes
  - Example of what happens when absolute performance is the goal
- Class 2 winners mostly PGAS models
  - Chapel received Class 2 recognition in five of ten years

Chapel and HPCC Awards

Chapel in 2006



Chapel in 2012

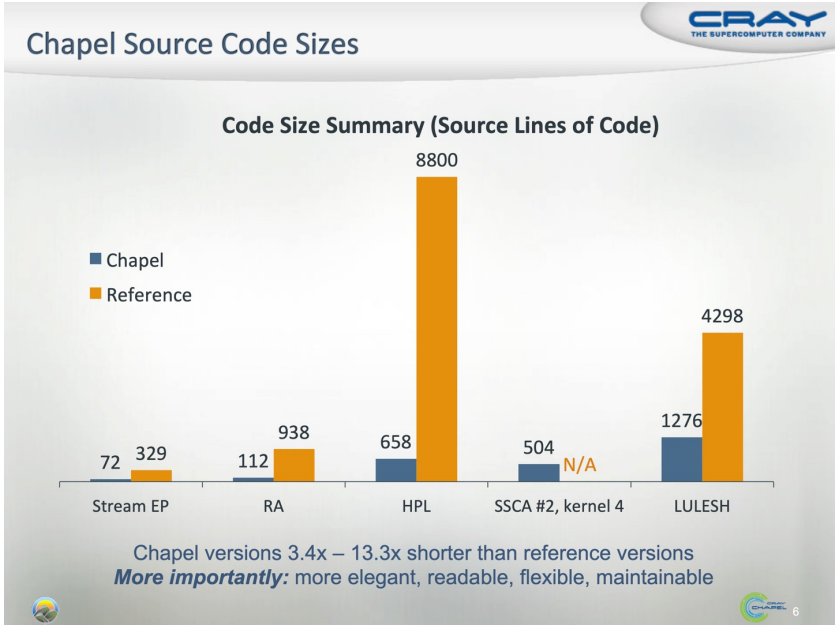


Image credits: Cray's slides at HPCC BOFs at SC06 and SC12
https://hpcchallenge.org/presentations/sc2006/chamberlain-slides.pdf
https://hpcchallenge.org/presentations/sc2012/ChapelHPCC2012.pdf

BERKELEY LAB

PGAS Publication Venues 2007 - present

- Conference on Partitioned Global Address Space Programming Models
  - 2007 - 2015, as PGAS'xy
- PAW / PAW-ATM https://go.lbl.gov/paw
  - Held annually at SC
  - 2016 & 2017: PGAS Applications Workshop
  - 2018: Parallel Applications Workshop, Alternatives To MPI
  - Since 2019: Parallel Applications Workshop, Alternatives To MPI+X
  - Current call: submissions due July 24

Outline

I. My early career or “why I drank the PGAS Kool-Aid”

II. The PGAS community 2001 to 2017

III. 2017 to the present

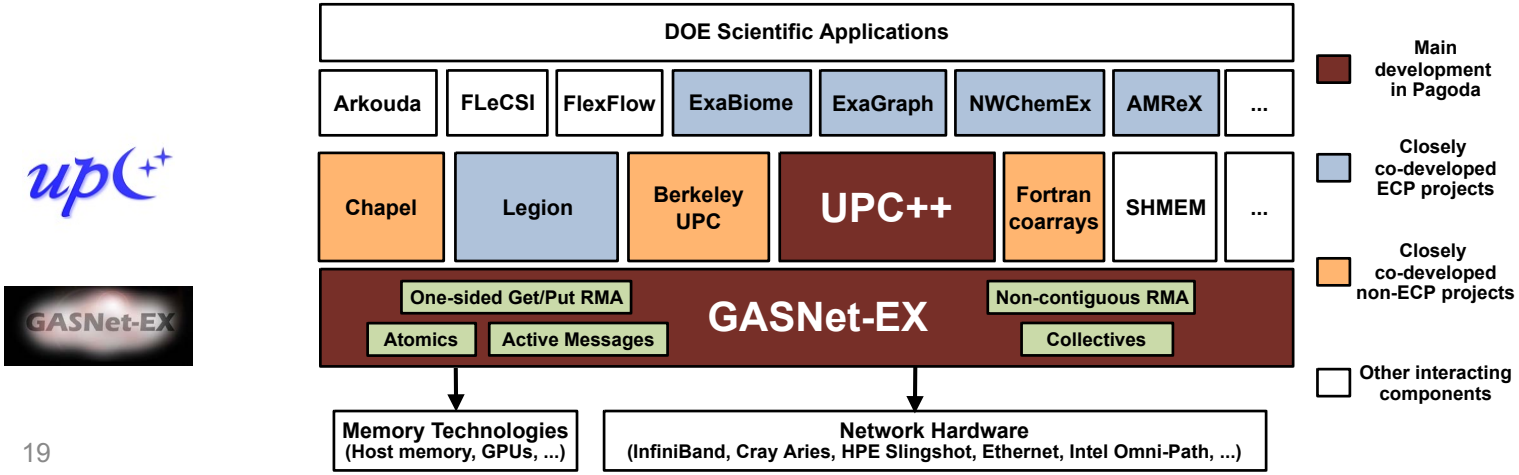
IV. Closing

The Pagoda Project

https://go.lbl.gov/pagoda

Support for lightweight communication for exascale applications, frameworks and runtimes

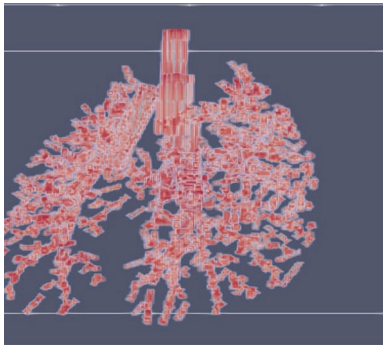
- **GASNet-EX** middleware layer providing a network-independent interface suitable for Partitioned Global Address Space (PGAS) runtime developers
- **UPC++** C++ PGAS library for application, framework and library developers, a productivity layer over GASNet-EX

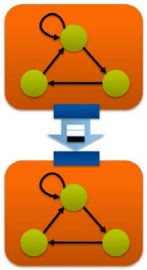


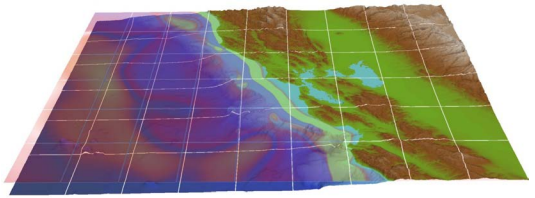
UPC++ Application Examples

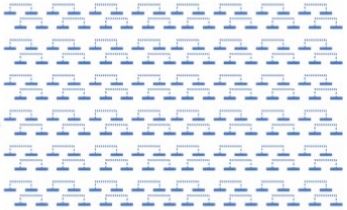
Several applications have been written using UPC++, resulting in improved programmer productivity and runtime performance. Examples include:

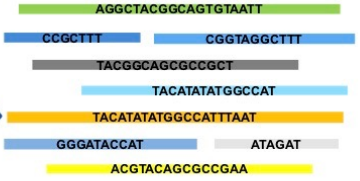
- Actor-UPCXX, used in the Pond tsunami simulator
- A UPC++ backend for NWChemEx/TAMM
- MetaHipMer, a genome assembler
- UPC++ DepSpawn, a library for data-flow computing
- Mel-UPX, half-approximate graph matching solver
- SIMCoV, agent-based simulation of lungs with COVID

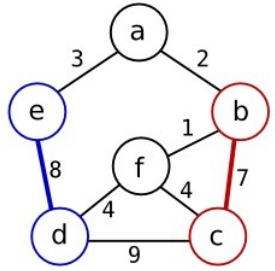










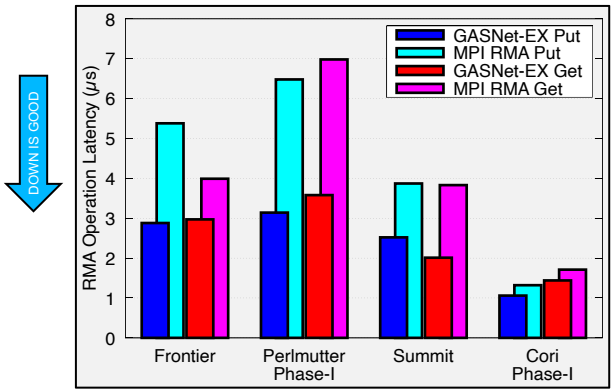


BERKELEY LAB

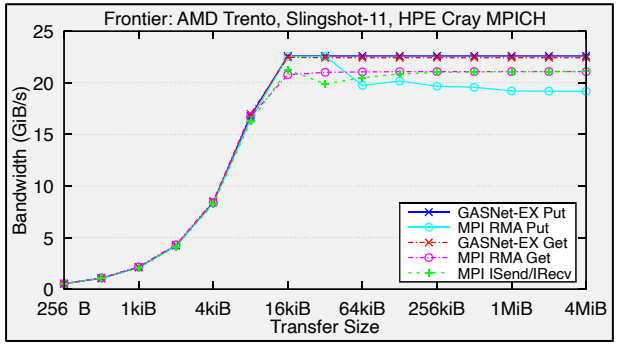
GASNet-EX RMA Performance versus MPI RMA and Isend/Irecv

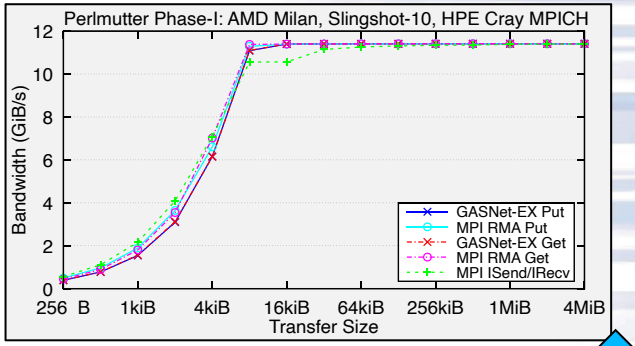
- Four distinct network hardware types
- The performance of GASNet-EX matches or exceeds that of MPI RMA and message-passing:
  - 8-byte Put latency 19 - 52% better
  - 8-byte Get latency 16 - 49% better
  - Better flood bandwidth efficiency: often reaching same or better peak at ½ or ¼ the transfer size

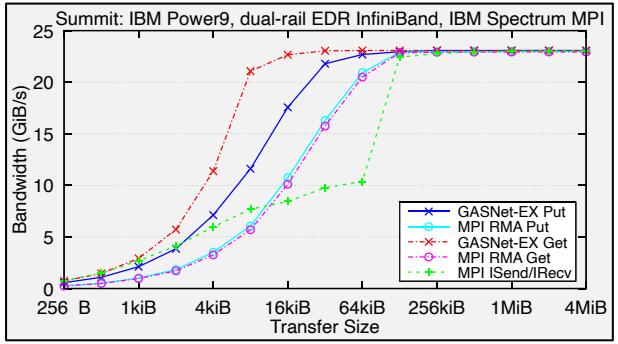
8-Byte RMA Operation Latency (one-at-a-time)

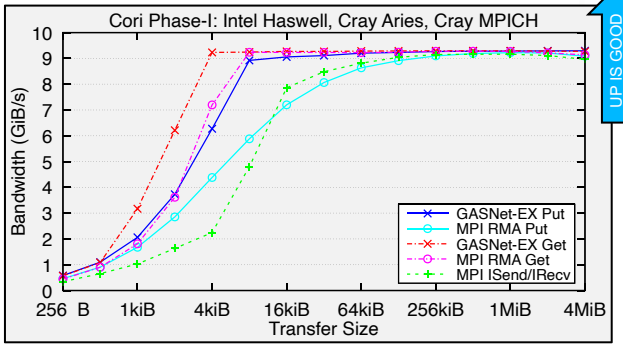


Uni-directional Flood Bandwidth (many-at-a-time)

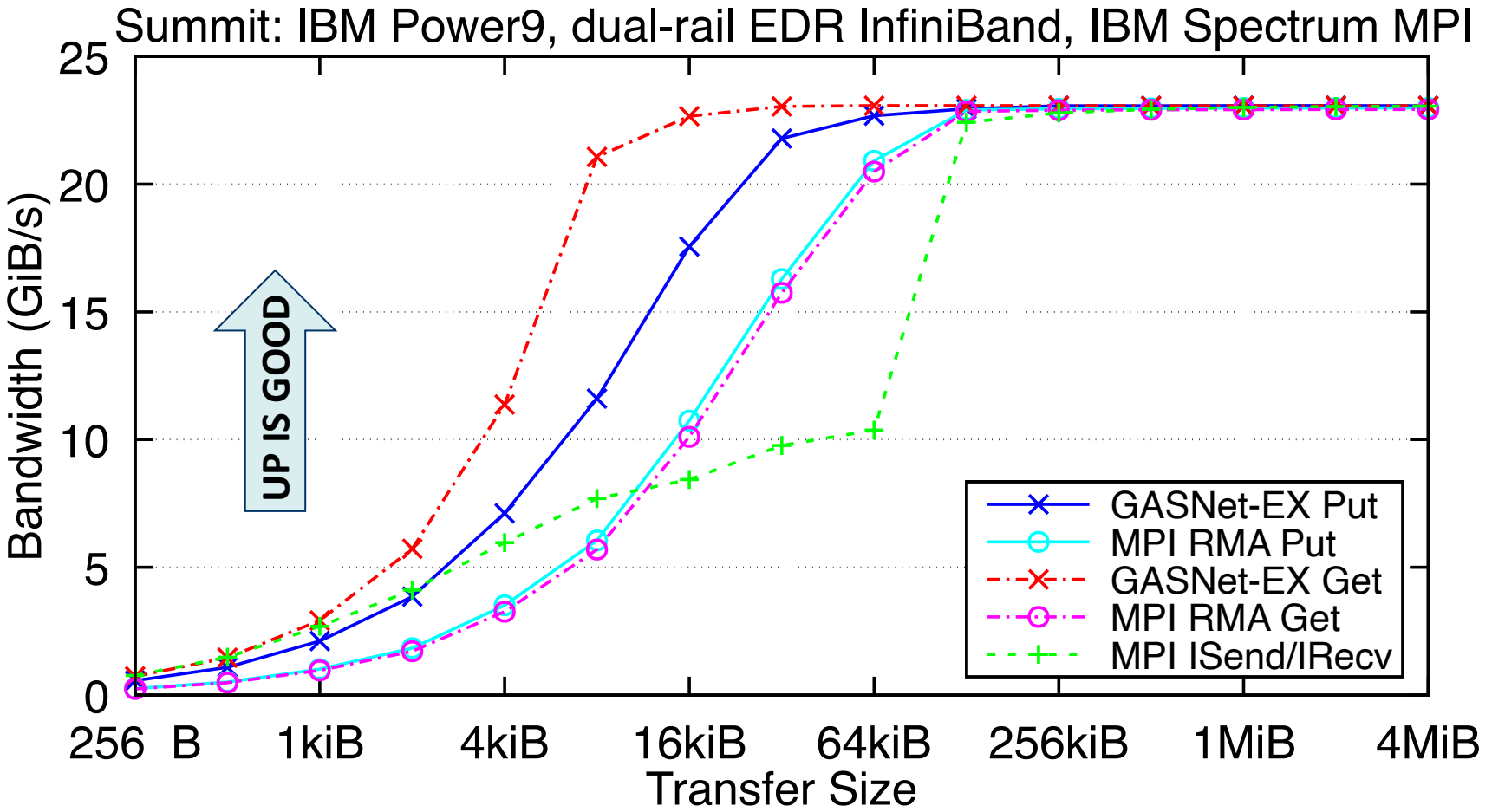








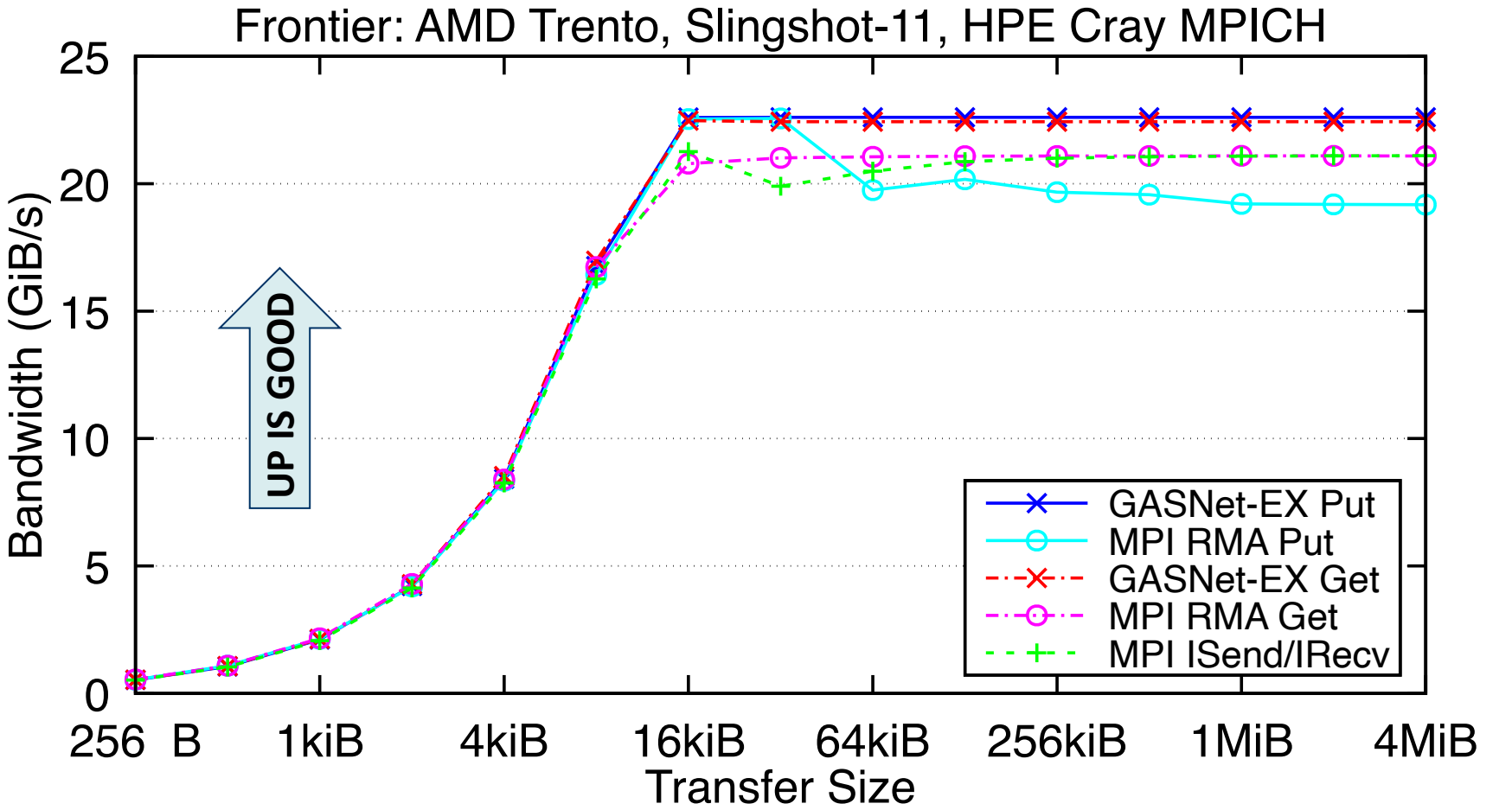
Perlmutter Phase-I results collected July 2022, all others collected April 2023.
GASNet-EX tests were run using then-current GASNet library and its tests.
MPI tests were run using then-current center default MPI version and Intel MPI Benchmarks.
All tests use two nodes and one process per node.
For details see LCPC'18 doi.org/10.25344/S4QP4W and PAW-ATM'22 doi.org/10.25344/S40C7D
See also: gasnet.lbl.gov/performance



#5 System on June 2023 Top500

A comparison of uni-directional point-to-point host-memory flood bandwidth benchmarks, run April 2023 on OLCF's Summit system.
Shows the performance of RMA (Put and Get) operations using GASNet-EX and both RMA and message-passing (Isend/Irecv) using IBM Spectrum MPI.
Results were obtained using current GASNet tests and Intel MPI Benchmarks, respectively.

BERKELEY LAB



#1 System on June 2023 Top500

A comparison of uni-directional point-to-point host-memory flood bandwidth benchmarks, run April 2023 on OLCF's Frontier system.
Shows the performance of RMA (Put and Get) operations using GASNet-EX and both RMA and message-passing (Isend/Irecv) using HPE Cray MPI.
Results were obtained using current GASNet tests and Intel MPI Benchmarks, respectively.

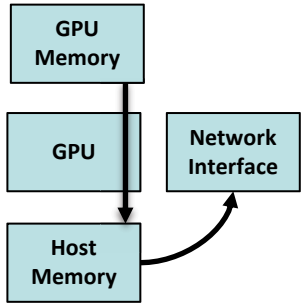
BERKELEY LAB

Accelerated RMA to/from GPU memory

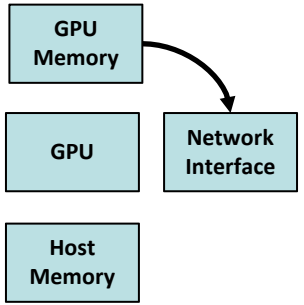
Modern GPUs and NICs can support peer-to-peer data transfers

Example: Put with source on GPU

- In the absence of necessary hardware and OS support:
  1. Data must be copied from GPU memory to host memory
  2. RDMA from host memory's copy
- With support:
  1. RDMA directly from GPU memory (no copies)



Data movement without acceleration



Data movement with acceleration

Accelerated RMA to/from GPU Memory

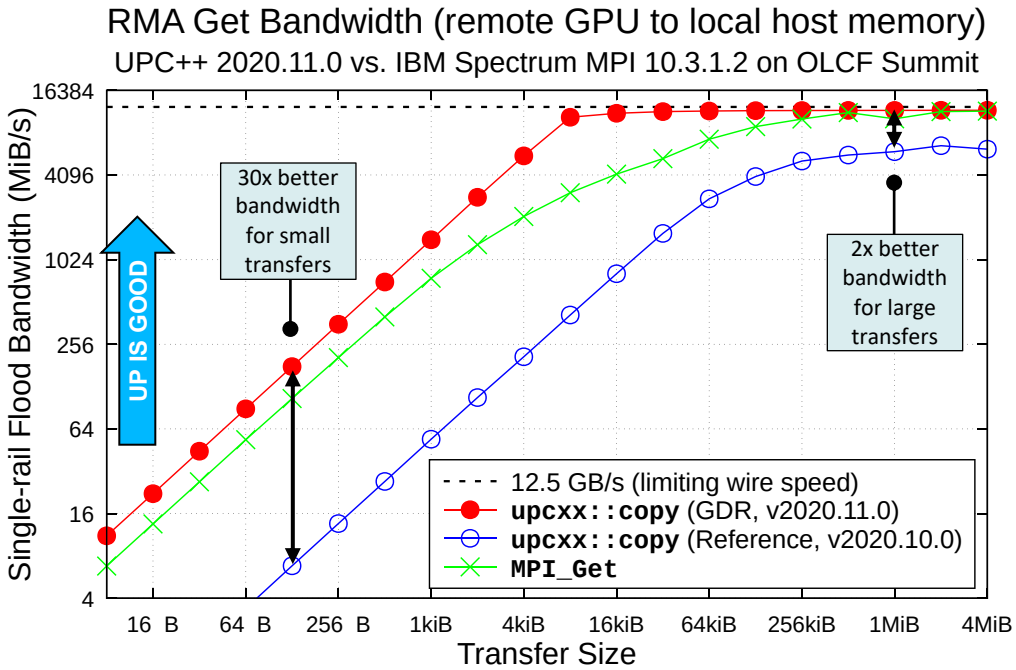
Measurements of flood bandwidth of **`upcxx::copy()`** on OLCF's Summit

Difference between two consecutive releases shows benefit of GASNet-EX's support for accelerated transfers via Nvidia's "GDR".

- No longer staging through host memory
- Large xfers: 2x better bandwidth
- Small xfers: up to 30x better bandwidth

Get operations to/from GPU memory now perform comparably to host memory

Comparisons to MPI RMA in GDR-enabled IBM MPI show UPC++ saturating more quickly to the peak



GASNet-EX

upc++

The Ohio State University

IBM Spectrum MPI

OAK RIDGE National Laboratory | LEADERSHIP COMPUTING FACILITY

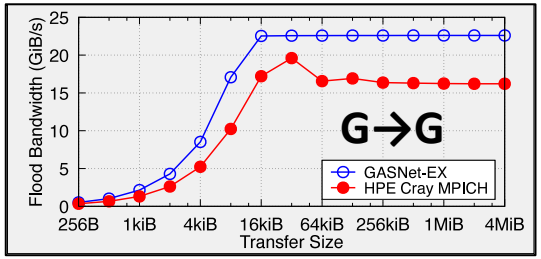
UPC++ results were collecting using the version of the **`cuda_benchmark`** test that appears in the 2020.11.0 release.
MPI results are from **`osu_get_bw`** test in a CUDA-enabled build of OSU Micro-Benchmarks 5.6.3.
All tests were run on OLFC Summit, between two nodes with one process per node, over its EDR InfiniBand network.

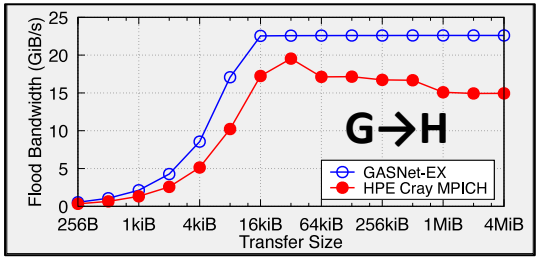
BERKELEY LAB

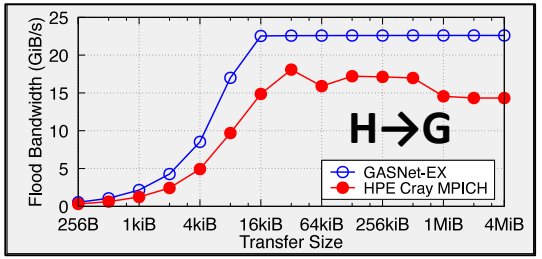
GPU memory RMA on OLCF's Frontier

Recent comparison of GASNet-EX and Cray MPICH performance on internode flood bandwidth benchmarks for six distinct combinations of (H)ost versus (G)PU memory and direction of transfer (Put or Get)

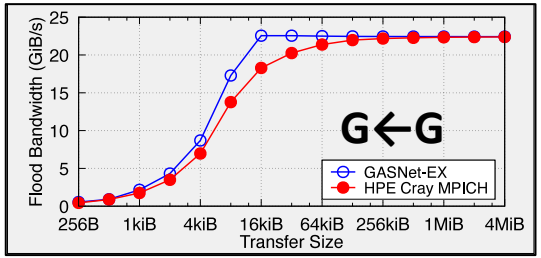
Put:

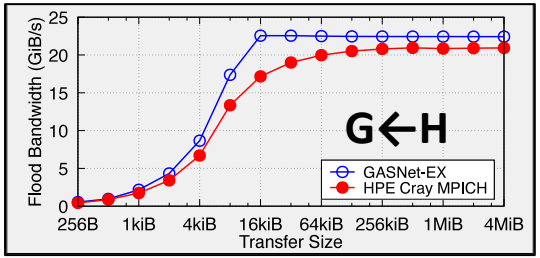


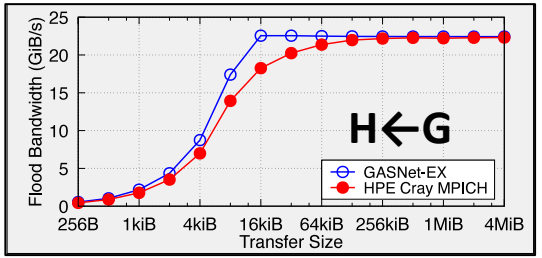




Get:







GASNet results were collecting using the **`testlarge`** benchmark that appears in the 2023.3.0 release.
MPI results are from **`osu_put_bw`** and **`osu_get_bw`** tests in a ROCM-enabled build of OSU Micro-Benchmarks 7.1-1.
All tests were run on OLCF Frontier in April 2023, between two nodes with one process per node, over its Slingshot-11 network.

Informal Survey of GPU RMA Support

	G→G	G→H	H→G	G←G	G←H	H←G
GASNet-EX (IB, SS10, SS11)	✔	✔	✔	✔	✔	✔
Cray MPICH (SS11)	✔	✔	✔	✔	✔	✔
Cray MPICH (SS10)	✘	✘	✔	✘	✔	✘
IBM Spectrum MPI (IB)	✘	✘	✘	✘	✘	✔
MVAPICH2-GDR (IB)*	✔	✔	✔	✔	✔	✔

Looking at three networks representative of top DOE systems.
Each ✔ or ✘ represent the (in)ability to complete the benchmarks by the best-behaving example of a given MPI *found by the author on DOE systems* in April and May 2023.

* Not provided by the center or vendor. Used OSU's binary distribution for Summit.

Outline

I. My early career or “why I drank the PGAS Kool-Aid”

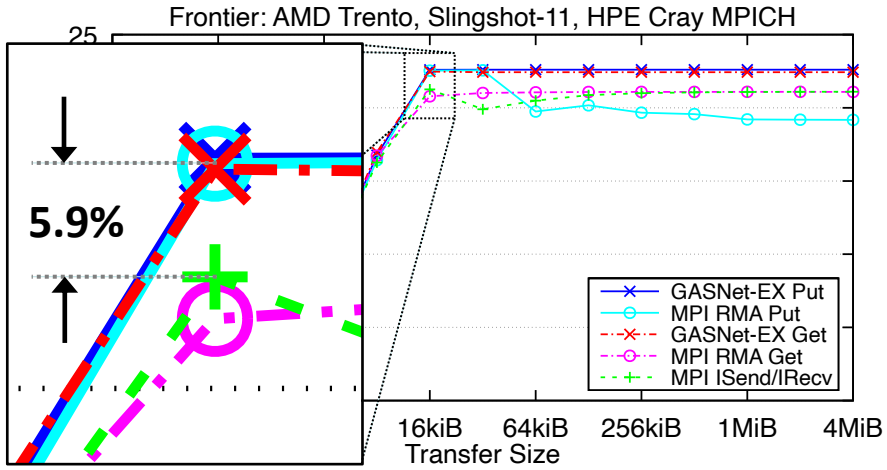
II. The PGAS community 2001 to 2017

III. 2017 to the present

IV. Closing

Thoughts on the Future (1 of 2)

- GASNet has helped programming model developers show that one need not sacrifice high performance to achieve high productivity
  - A good semantic fit to network capabilities has historically provided a performance advantage over message-passing
- However, modern systems have narrowed some of the performance gaps for host memory RMA
  - RMA versus message passing
  - GASNet versus MPI RMA



Thoughts on the Future (2 of 2)

- Current GASNet and UPC++ comparisons to MPI show there is *still* a network performance advantage for RMA to/from GPU memory
  - At least for now
- Nothing has changed in 20+ years to erode the productivity arguments for PGAS over message passing
  - And the prospects look good for extending this to GPUs as well
- I look forward to taking part in another decade of PGAS

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** (NERSC), a U.S. Department of Energy Office of Science User Facility located at Lawrence Berkeley National Laboratory, operated under Contract No. DE-AC02-05CH11231 using NERSC award DDR-ERCAP0023595..

This research used resources of the **Oak Ridge Leadership Computing Facility** at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

THANK YOU!

go.lbl.gov/paul-hargrove

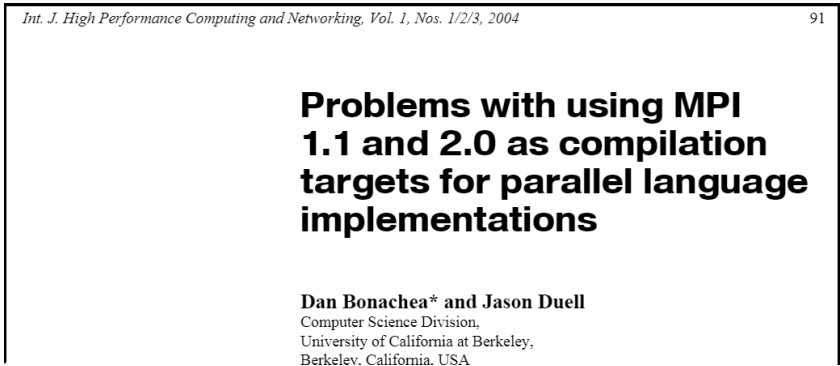
gasnet.lbl.gov

upcxx.lbl.gov

BACKUP SLIDES USED IN Q&A

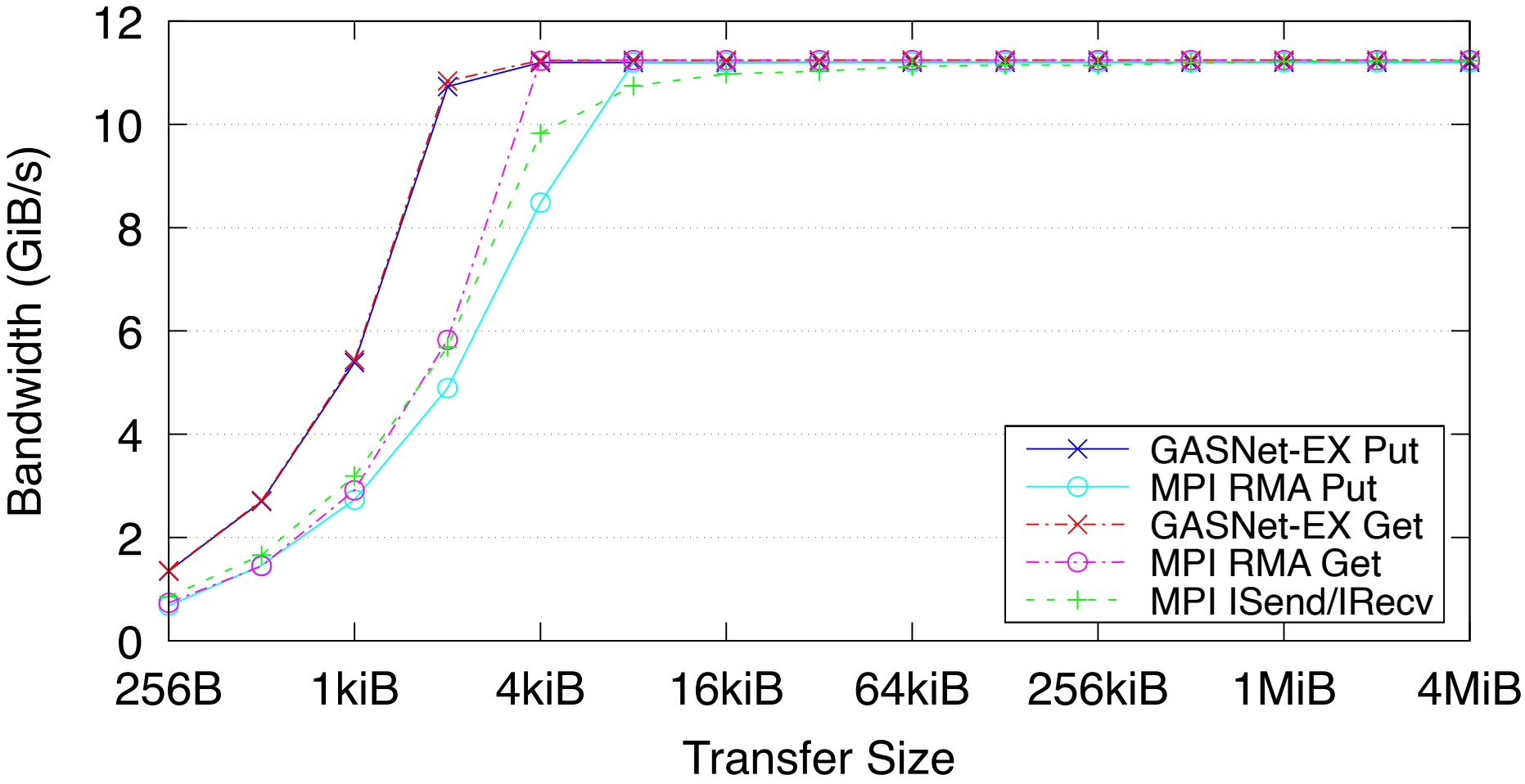
GASNet-1: Historical Overview

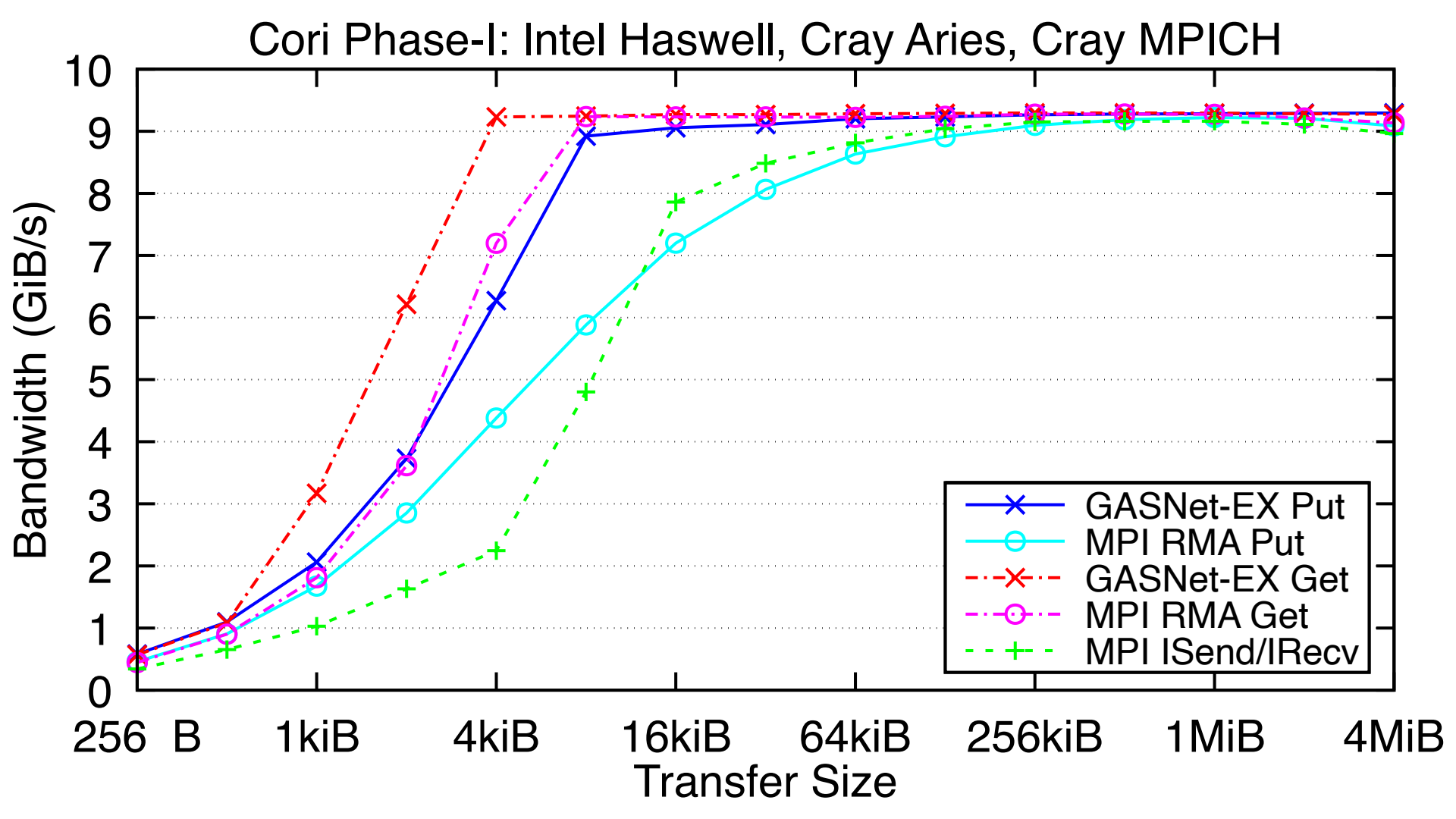
- Started in 2002 to provide a portable network communication runtime for three PGAS languages:
  - UPC, Titanium and CAF
- Primary features:
  - Non-blocking RMA (one-sided Put and Get)
  - Active Messages (simplification of Berkeley AM-2)
- Motivated by semantic issues in (then current) MPI-2.0
  - https://doi.org/10.1504/IJHPCN.2004.007569


Int. J. High Performance Computing and Networking, Vol. 1, Nos. 1/2/3, 2004 91

Problems with using MPI 1.1 and 2.0 as compilation targets for parallel language implementations

Dan Bonachea* and Jason Duell
Computer Science Division,
University of California at Berkeley,
Berkeley, California, USA


JLSE Arcticus: Intel Ice Lake, EDR InfiniBand, Intel MPI
Bandwidth (GiB/s)
0
2
4
6
8
10
12
256B
1kiB
4kiB
16kiB
64kiB
256kiB
1MiB
4MiB
Transfer Size
GASNet-EX Put
MPI RMA Put
GASNet-EX Get
MPI RMA Get
MPI ISend/IRecv


Cori Phase-I: Intel Haswell, Cray Aries, Cray MPICH
Bandwidth (GiB/s)
0
1
2
3
4
5
6
7
8
9
10
256 B
1kiB
4kiB
16kiB
64kiB
256kiB
1MiB
4MiB
Transfer Size
GASNet-EX Put
MPI RMA Put
GASNet-EX Get
MPI RMA Get
MPI ISend/IRecv