12th ANNUAL WORKSHOP 2016

GASNet: Global Address Space Networking

Paul H. Hargrove

Lawrence Berkeley National Lab

April 8th, 2016

OUTLINE

- Background
- Early IB (VAPI) Support
- Support for Current OFA APIs
- Work-in-progress and Future Work

BACKGROUND

PARTITIONED GLOBAL ADDRESS SPACE (PGAS)

https://en.wikipedia.org/wiki/Partitioned_global_address_space

- **PGAS is a programming model**
  - An alternative to message-passing (MPI) for writing distributed applications
  - Encompasses many languages and libraries (partial list on the next slide)
    - Examples: **UPC++** and **OpenSHMEM** are the subjects of the next two talks
  - Shared-memory style programming in a distributed-memory environment
    - UPC pointer-to-shared, Fortran 2008 Coarrays, etc.
  - Global memory is partitioned into portions with affinity to each thread
  - Affinity-awareness aides locality of reference (communication avoidance)

- **Asynchronous PGAS (APGAS)**
  - Extends PGAS with local and remote asynchronous task creation
    - Examples: Cray Chapel and IBM X10

GASNet

http://gasnet.lbl.gov

- **GASNet is "Global Address Space Networking"**
  - A communications library for implementing PGAS/APGAS languages and libraries
    - Main focus is on RMA (Put/Get) and RPC (Active Messages)
  - A project of Lawrence Berkeley National Laboratory (LBNL) and the University of California Berkeley (UCB), begun in 2002 to support UPC and Titanium
  - Runs on everything from laptops to supercomputers
- **GASNet has become the *de facto* standard in its field, with projects using it for their communications including:**
  - Unified Parallel C (a.k.a. "UPC")
    - Berkeley UPC (LBNL and UCB)
    - GNU UPC (Intrepid Technology)
    - Clang UPC (Intrepid Technology)
    - UPC for Cray XT (Cray)
  - Fortran 2008 Coarrays
    - OpenUH Fortran compiler (UH)
    - OpenCoarrays for gfortran
    - CAF for Cray XT (Cray)
  - CAF 2.0 (Rice)
    - A superset of Fortran 2015
  - OpenSHMEM (UH and ORNL)
    - Reference implementation
  - Legion (Stanford)
  - UPC++ (LBNL)
  - Habanero-UPC++ (Rice and LBNL)
  - Global Arrays / NWChem (LBNL)
    - Emerging prototypes
  - Titanium (UCB)
  - Cray Chapel (Cray)
  - And more…

GASNet API

- **GASNet "Core API"**
  - Active Message (Remote Procedure Call) interfaces
    - An AM Request invokes a registered "handler" function in target process
    - Request handler may send an optional Reply, but no other comms allowed
  - Minimum requirement for a new network port
- **GASNet "Extended API"**
  - Remote Put and Get operations
    - One sided: caller provides all address and length information
    - Blocking operations
    - Implicit (region-based) and Explict (handle-based) non-blocking operations
  - Split-phase Barrier
  - "Reference Extended" implements the Extended API in terms of the Core API
    - Typically a network-specific "native" implementation will replace the reference one
- **Extensions (non yet in the official specification)**
  - Collectives (non-blocking by default)
  - Vector/Index/Strided operations (a.k.a. VIS)

EARLY IB (VAPI) SUPPORT

GASNet OVER LIBVAPI

- **GASNet's first InfiniBand support was vapi-conduit** *
  - Used Mellanox's VAPI (Verbs API)
  - First appeared in October 2003 release
  - Retired in October 2013 release
- **GASNet Core API (Active Messages)**
  - Primary path based on SEND_WITH_IMM
    - Credit-based flow control (never see RNR)
  - Secondary path uses RDMA_WRITE for small number of "hot peers"
    - Poll in memory for message arrival (instead of CQ)
- **GASNet Extended API (RMA operations)**
  - Common/preferred case has destination in memory registered at initialization
  - Operations are mapped to RDMA_READ and RDMA_WRITE operations
  - wr_id field connects CQE back to GASNet-level operation
  - Inline send used when possible
  - Uses own counters (semaphore semantics) to track SQ/CQ depth

* "conduit" is GASNet's term for network-specific support code.

DYNAMIC MEMORY REGISTRATION

- **The GASNet "Segment"**
  - All remote addresses must lie in a "segment" established at initialization time
  - In "FAST" mode GASNet pre-registers the segment memory and shares the Rkeys
    - Segment size is limited by what can be registered
  - In "EVERYTHING" mode the segment spans all of memory
    - Cannot pre-register with the HCA is this case
- **Dynamic registration used when src or dst not pre-registered**
  - FAST mode: local address not in the pre-registered segment
  - EVERYTHING mode: all local and remote addresses
  - However, may use immediate send or bounce buffers locally for small transfers
- **The "firehose" library for caching of memory registration***
  - Not specific to InfiniBand
  - Exposes one-sided, zero-copy RDMA as the common case
  - Degrades gracefully to rendezvous as working set grows
  - Susceptible to errors when cached memory is unmapped (will come back to this)

* C. Bell and D. Bonachea. "A New DMA Registration Strategy for Pinning-Based High Performance Networks". Workshop on Communication Architecture for Clusters (CAC'03), 2003.

ASYNCHRONOUS PROGRESS

- **Active Message handlers must run on a host CPU**
  - Synchronously when code calls GASNet_AMPoll()
  - Asynchronously (optional) using a progress thread
- **In general non-blocking RMA ops proceed asynchronously**
  - However, in EVERYTHING mode firehose may require an Active Message round-trip on a cache-miss for a remote address
- **Vapi-conduit AM-progress thread**
  - Used EVAPI_set_comp_eventh()/EVAPI_poll_cq_block()
  - Failed to find "well-behaved" applications that would benefit
  - Applications with good "network attentiveness" saw performance *decline*
    - Contention (lock, cache, etc.) between application and progress thread

SUPPORT FOR CURRENT OFA APIS

GASNet OVER LIBVERBS

- **GASNet's "ibv-conduit"**
  - First appeared in October 2007 release
  - Originally shared its source code with vapi-conduit
    - Use of #ifdef and typedef as necessary
    - This dual-conduit code base ended with retirement of vapi-conduit in 2013
  - Fundamentals of the design have not changed since libvapi
- **Implementation as evolved significantly over time**
  - SRQ – found this was an absolute necessity at large scale
    - Prior to SRQ, runs with 4K processes could post nearly all of memory to RQ
  - XRC – also critical to operation at large scale
    - Prior to XRC, runs with 64K processes not possible w/ only 64K QPs/HCA
    - On 16-core node use of XRC yields 256-fold reduction in QP usage
    - Currently supporting both Mellanox and OFED APIs for XRC
  - Lazy-connect (optional) as another way to reduce memory and QP consumption
  - Asynchronous progress
    - Now use ibv_create_comp_channel()/select()/ibv_get_cq_event()
    - Today see actual performance benefit from use of a progress thread

GASNet OVER LIBFABRIC

- **GASNet's "ofi-conduit"**
  - Contributed by Intel Corporation
  - First appears in October 2015 release
  - Subject of PGAS 2014 paper* (when still known as "sfi-conduit") showing a measurable reduction in cycle counts relative to ibv-conduit
  - Simple LoC metric says 80% smaller than ibv-conduit
- **GASNet Core API (Active Messages)**
  - Reliable datagram endpoint
  - Required capabilities: FI_MSG and FI_MULTI_RECV
- **GASNet Extended API (RMA operations)**
  - Reliable datagram endpoint
    - EP and CQ distinct from Core API, only polled if RMA operations are in-flight
  - Required capabilities: FI_RMA
  - Segment registration: FI_MR_SCALABLE

* Miao Luo, Kayla Seager, Karthik S. Murthy, Charles J. Archer, Sayantan Sur, and Sean Hefty. "Early Evaluation of Scalable Fabric Interface for PGAS Programming Models." Proc. Eighth Conf. on Partitioned Global Address Space Programming Models (PGAS). Oct 2014.

WORK-IN-PROGRESS AND FUTURE WORK

GASNet-EX

Currently Funded Work

- **GASNet-EX modernizes GASNet for Exascale**
- **Incorporates 15 years worth of "lessons learned"**
- **Recognizes that requirements have changed significantly**
  - From few to hundreds of CPU threads per NIC
  - From modest to huge memory per node (and thus NIC)
  - From PGAS to Asynchronous PGAS (APGAS) languages
- **Major modernization themes include**
  - Standardize existing extensions to GASNet
  - Support multiple clients (e.g. hybrid apps)
  - Support resilient clients
  - Support threads as first-class entities
  - Better manage "time" (polling)
  - Better manage "space" (buffers)
  - Discard some legacy baggage

GASNet RESILIENCE

Currently Funded Work

- **APIs for building resilient PGAS runtimes**
  - A component of the GASNet-EX modernization effort
  - Have adopted an exception-based approach
    - Client runtimes can register callbacks to be invoked when errors occur
  - Are providing mechanism to recover from loss of processes, nodes, links, etc.
    - Intended to support client's implementation of any reasonable policy
- **Resilience of the GASNet implementation itself**
  - Could characterize as replacing "assert()" with "throw"
    - Except that GASNet is written in C, and we must therefore unwind manually
- **Checkpoint/restart support using BLCR***
  - Ibv-conduit support for collective checkpoint, restart and partial-rollback.
  - Will appear in the next GASNet release (late April or early May, 2016)
- **C/R work for non-collective consistent-state capture**
  - More challenging to reason about RMA/PGAS than about message passing
  - Seeking possible protocol or API enhancements at the level of Verbs or OFI

* BLCR = "Berkeley Lab Checkpoint/Restart", a kernel-level implementation of transparent process checkpoint/restart, also supported my several MPI implementations including MVAPICH2.

FUTURE WORK

- **Future work within ibv-conduit**
  - We are excited by ODP as an alternative to dynamic registration
    - Could replace 500 lines of code devoted to ibv-specific part of firehose
    - Correct behavior even in presence of munmap()
  - Enhance firehose to use ummunotify
    - Allow us to tolerate munmap() without ODP
  - Automatic Path Migration (APM)
    - Part of resilience work for recovery from link failures
  - Multicast
    - Collectives
    - Recovery code for resilience
- **Future work within ofi-conduit**
  - Checkpoint/restart support to match ibv-conduit's capacities
  - Too early to judge what else can/should be done
    - This presenter has so far only used the sockets provider

OPENFABRICS
ALLIANCE

12th ANNUAL WORKSHOP 2016

THANK YOU

Paul H. Hargrove

Lawrence Berkeley National Lab