OPENFABRICS ALLIANCE

2013 OFA Developer Workshop

#OFADevWorkshop

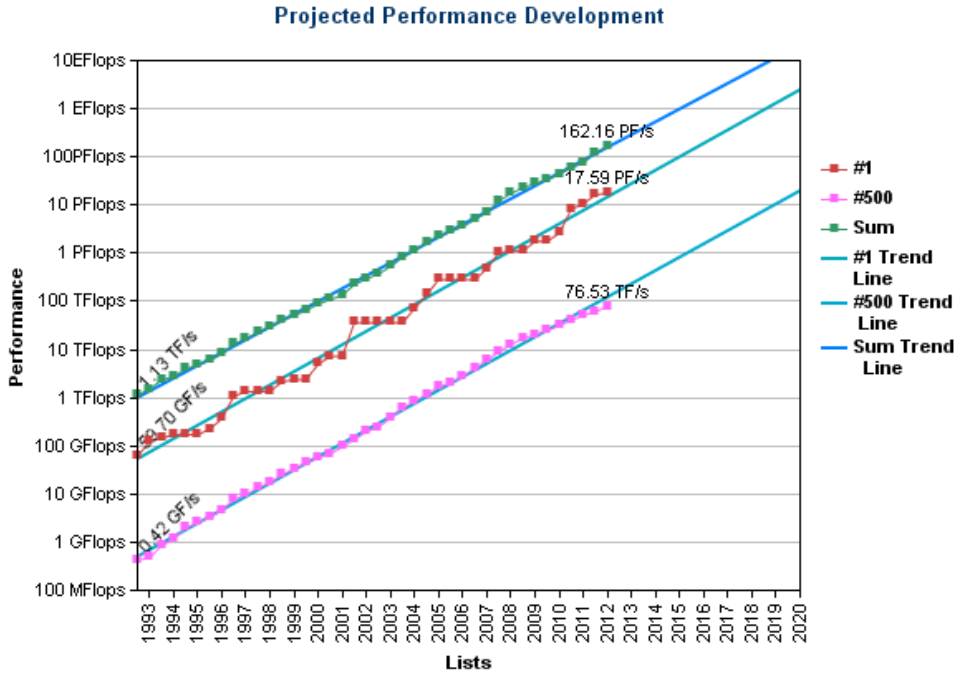
Challenges of Scale and APIs for MPI and PGAS

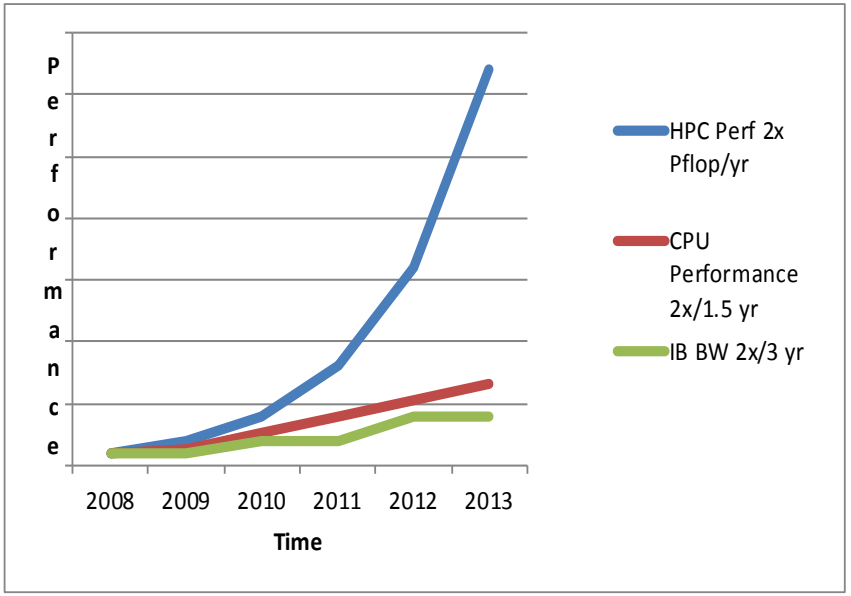
Author: Todd Rimmer
Date: April 2013

Agenda

- Projected HPC Scalability Requirements
- MPI/PGAS API Needs
- Management Traffic
- Near Term Improvements

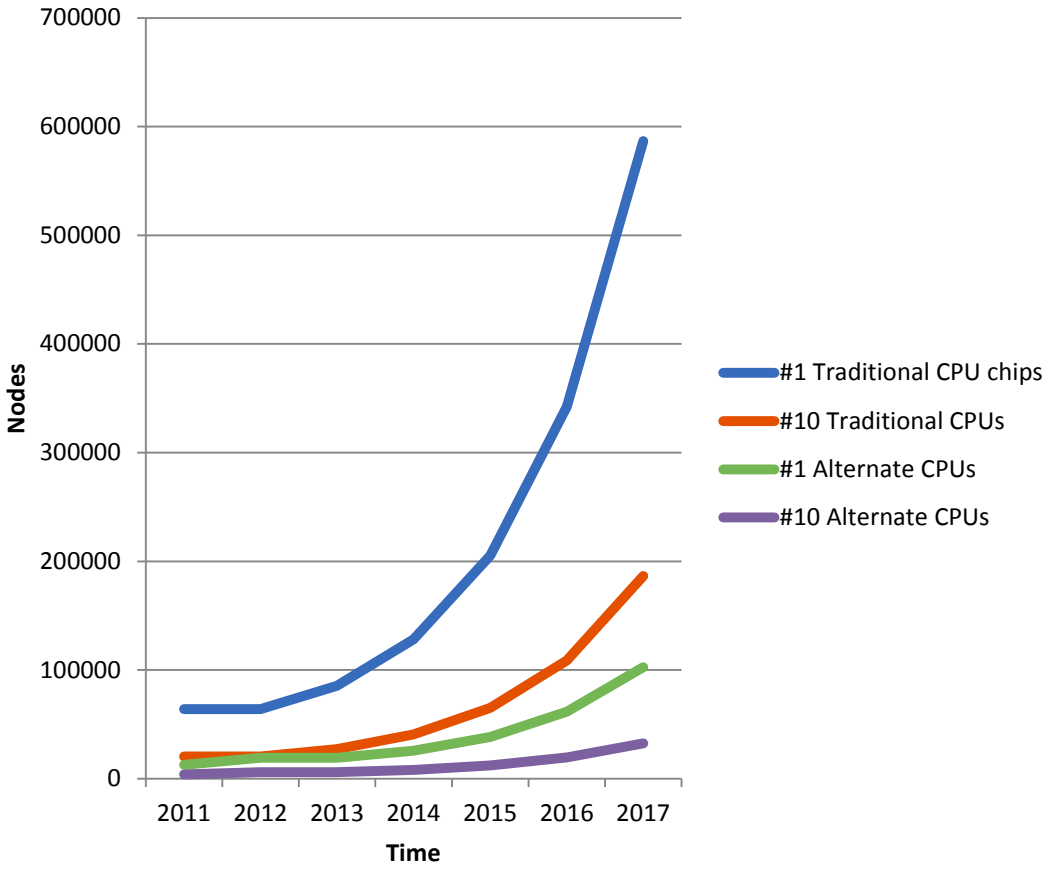
Projected HPC Scalability Requirements





- HPC Requirements are Outpacing Moore's Law
- Outpacing IB performance growth

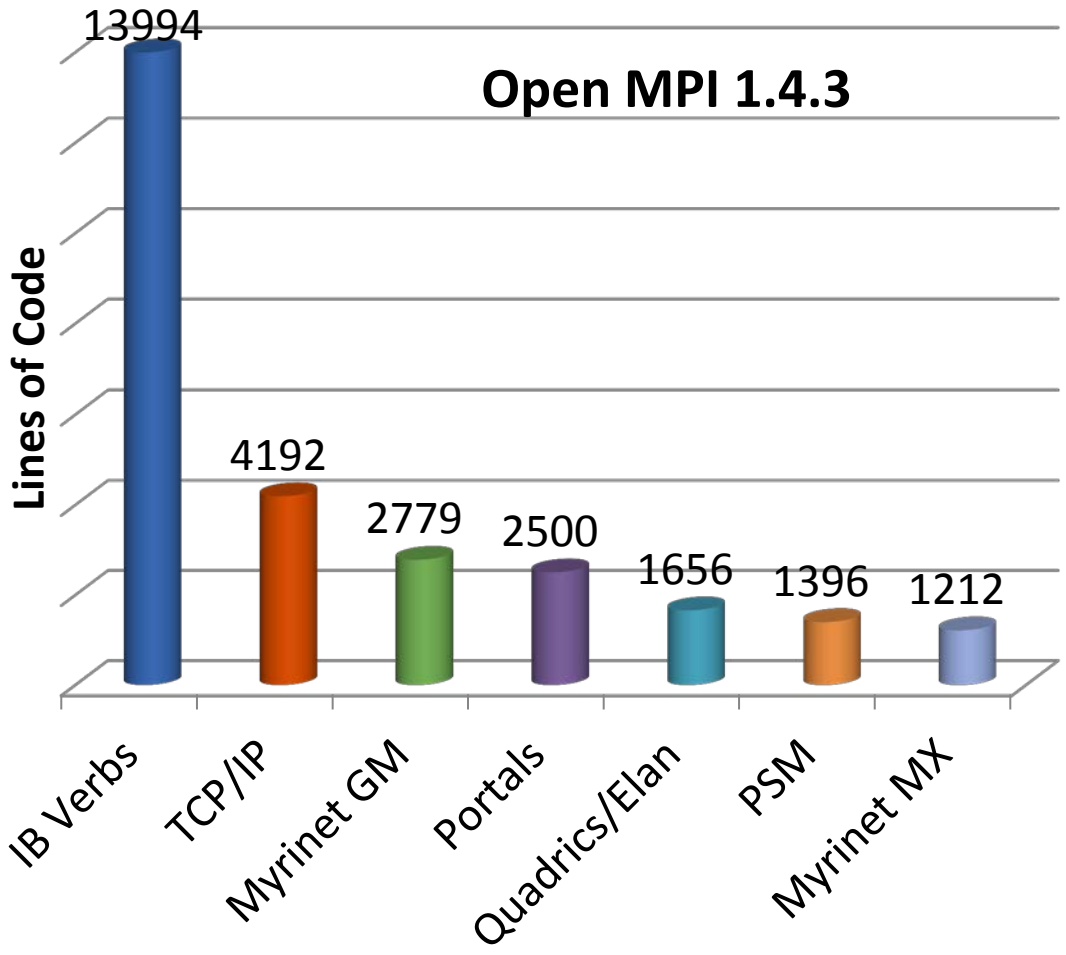
Projected HPC Scalability Requirements



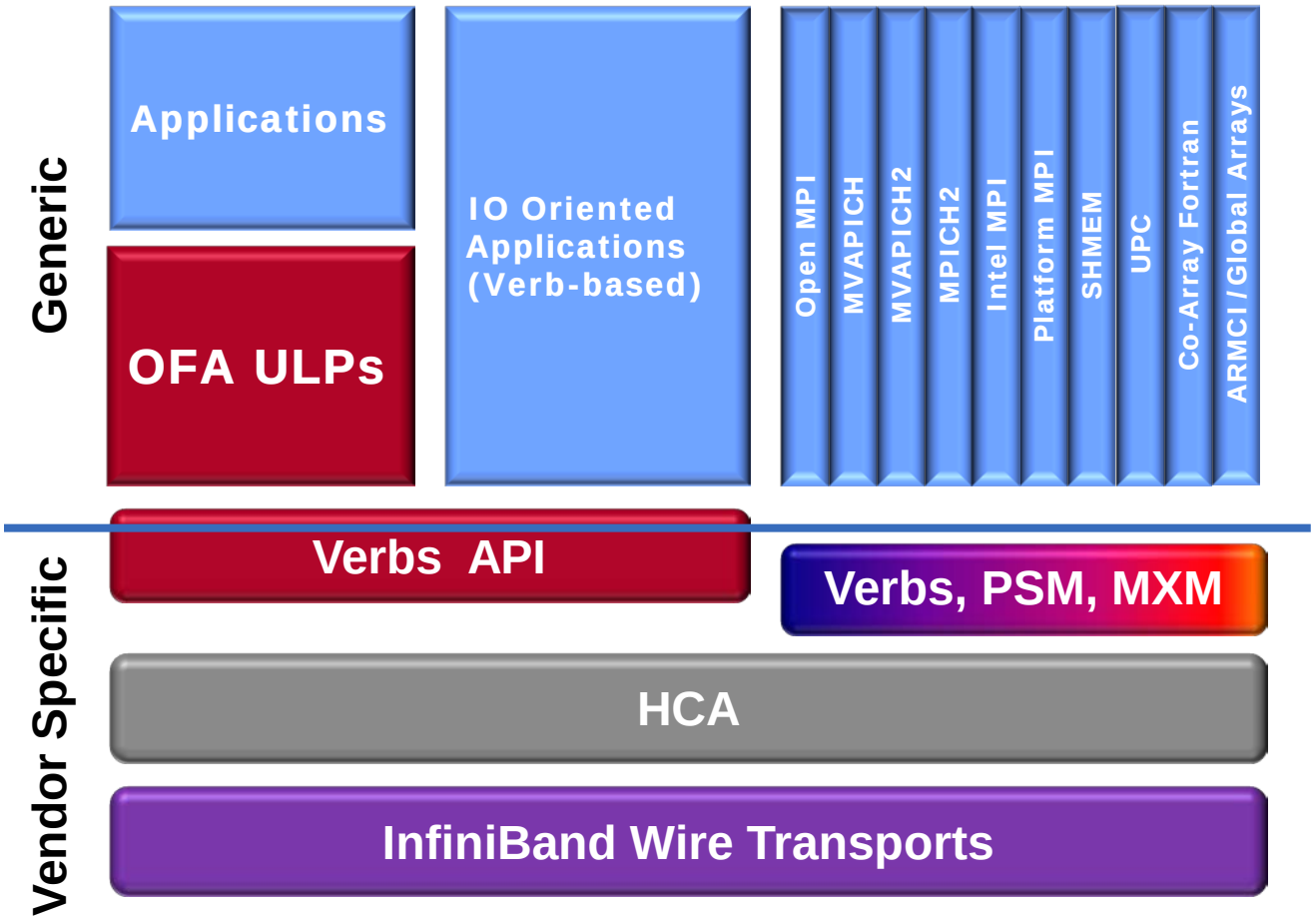
- Result is Rapidly increasing node counts
- Due to slower pace of interconnect speed growth
  - need multi-rail clusters
  - HCA counts will grow even faster

Comparison of Impedance Match OpenMPI MTL and BTL sizes

- Verbs is a bad match for MPI
  - Semantic mismatch, connected mode scalability, etc.
- HPC focused interconnects are a better fit
  - Such as PSM, Quadrics, Myrinet
- Relative sizes are similar for other MPIs
  - mvapich, mvapich2, etc.



How OFA Stack has Evolved



Requirements for Compute

- **Focus on the needs of MPI, PGAS and HPC Compute**
- **Design for very high HPC messaging rate, scalable latency up to Exascale cluster sizes**
  - Low overhead APIs
- **Maintain a minimal memory footprint**
  - Minimal memory footprint per end point
  - Scale out to large job size in support of Exascale
- **Support needs of multiple MPI and PGAS Middlewares**
  - Close alignment with variety of “channel interfaces”
  - Avoid burdening middleware with interconnect details
- **Support multiple hardware vendors**
  - Allow for hardware vendor integration
  - Offloads, Collectives, protocol optimizations

Avoid Middleware Complexity Delegate below Middleware

- MPI tag matching
- Optimization of data movement
  - Point to point: eager, rendezvous, etc.
  - Collectives
- Path Resolution & End Point Establishment
  - Multi-Rail
  - Dispersive routing
- Protocol Details
  - Resiliency algorithms
  - Memory locking
  - QoS

Key Mgmt Scalability Bottlenecks

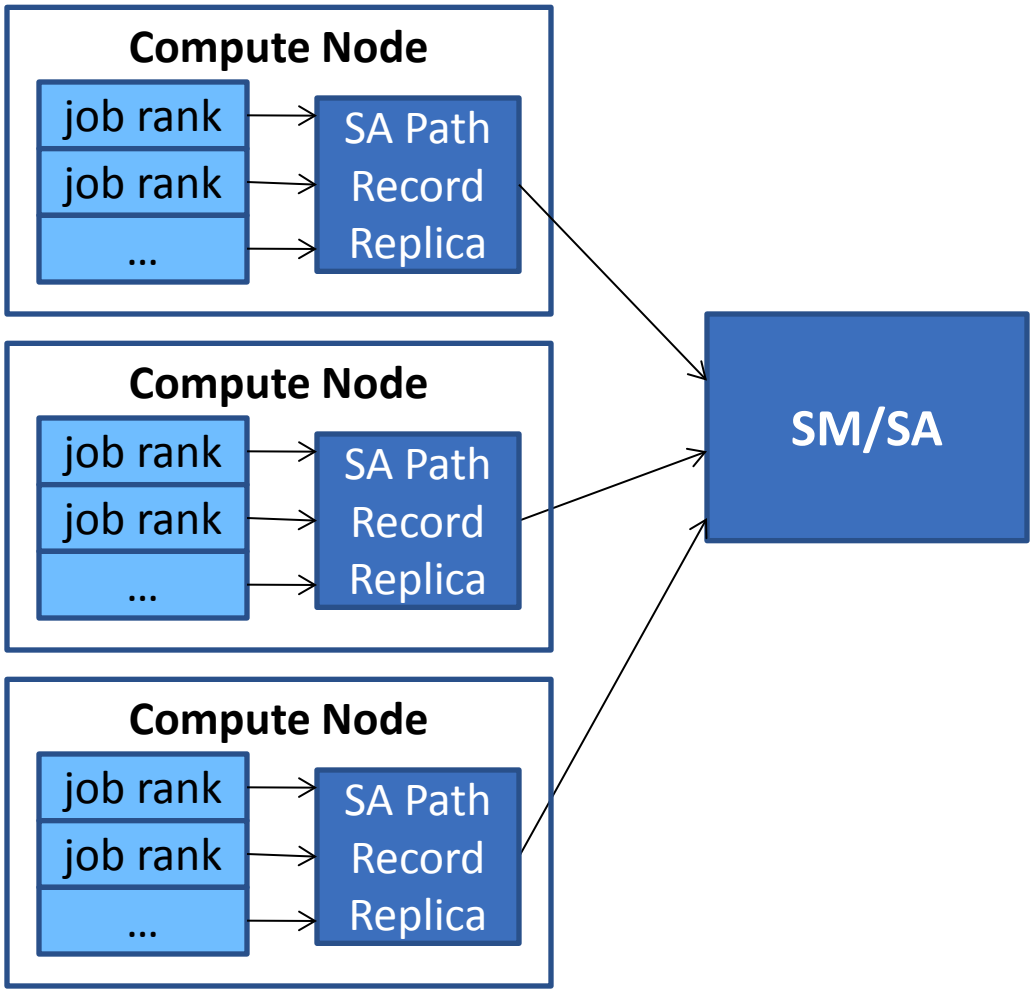
- PathRecord Query
- SA Query
- IPoIB ARP

PathRecord Query

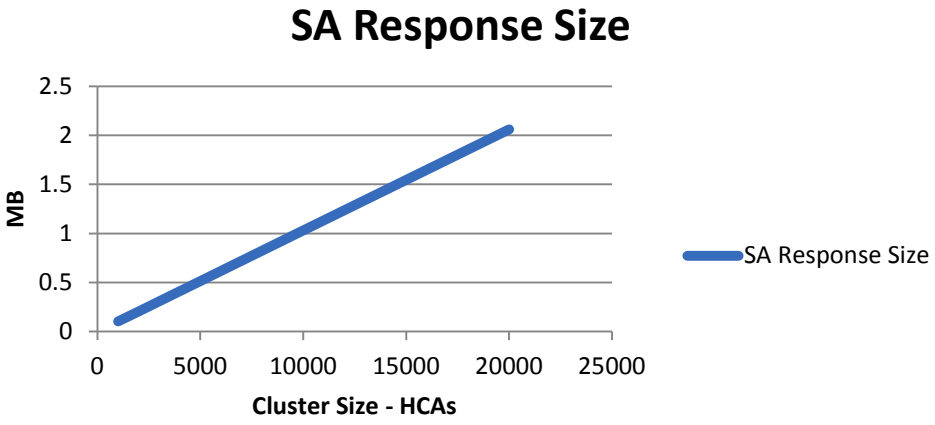
- Need a multi-tiered approach
  - Small clusters can do direct PathRecord query
  - Modest clusters can do PathRecord caching
  - Large clusters need PathRecord replicas or other techniques
  - Huge clusters need algorithmic approaches
    - Topology dependent
- Need to 1st standardize a plug-in API
- Need all ULPs, benchmarks, demos, diagnostics, CM etc. to use the API
  - Both kernel and user space
- Implement direct and cached plugins to start

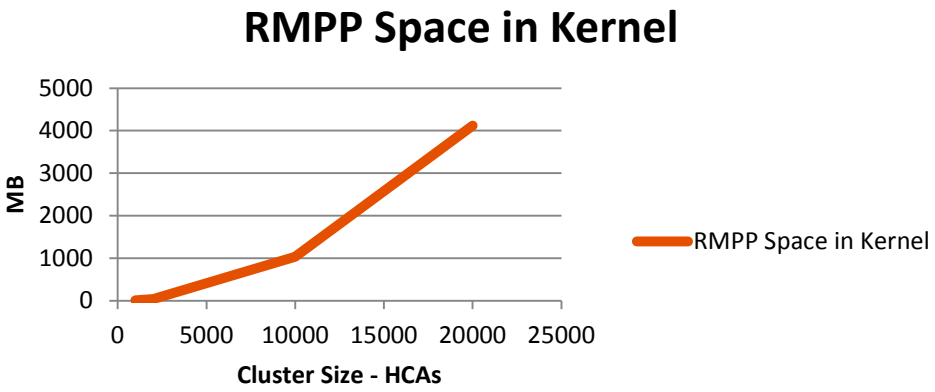
Scalable Path Resolution

- Each node retains and synchronizes a PathRecord replica with the SM/SA
  - Automatic update on fabric change
- Replica persists beyond life of jobs
  - Shared by all ranks on node
- Replica allows >1 Million PathRecord query/sec per node
- Permits very rapid job startup and avoids SA being a bottleneck in large fabrics



Management Traffic SA Query





- Assumption – concurrently many nodes do a O(HCAs) query
- This results in O(HCAs^2) growth in kernel memory
- Actual Growth can be worse due to increased overlap of larger responses

Near Term Improvements
RMPP Server Scalability

- RMPP handling in kernel makes sense for clients
  - Simplifies client APIs and implementation

- RMPP Server Handling in kernel is an SM/SA bottleneck
- Causes exponential growth in kernel memory use for large clusters
- Prevents sophisticated optimizations such as:
  - Response buffer reuse/sharing by SM/SA to reduce memory footprint
  - Response buffer pacing
  - Window size fine tuning per client

- With very minor changes, RMPP Server side handling can be optionally handled in application space

Near Term Improvements
SA Query Scalability

- SA Response Timeout/Retry Handling
  - Client uses fixed timeouts
  - Timeouts chosen a priori without knowledge of SA nor fabric load
- Need centralized config of timeouts and retry settings
  - As opposed to per application constants
- Retries should perform non-linear backoff

- SA Busy Response Handling
  - Present OFA code does immediate retry
  - Prevents SA from using BUSY to pace its workload
  - SA forced to discard
- BUSY should cause client backoff before attempting retry
  - Non-linear backoff also recommended

IPoIB ARP Scalability

- Need a multi-tiered approach in IPoIB
  - Modest clusters can do standard ARP/broadcast
  - Large clusters need pre-loaded ARP tables
  - Huge clusters need algorithmic approaches
    - Topology dependent
- Need to 1st standardize a plug-in API
- API needs to tie into PathRecord Plug-In
- Implement standard ARP and pre-loaded plugins to start

Summary

- HPC cluster sizes will grow year over year
- Compute stacks are becoming vendor specific
- OFA implementation of IBTA mgmt will be a bottleneck
- Some near term improvements are available
- Long Term solutions need flexibility via Plug-Ins

OPENFABRICS ALLIANCE

Thank You

Author: Todd Rimmer
Date: April 2013