Unified MPI and PGAS (UPC, OpenShmem, etc.)
Design with RDMA to Support Hybrid Programming
Environment for Exascale Systems

A Presentation at OFA Conference, Monterey 2012
by

Dhabaleswar K. (DK) Panda
The Ohio State University
E-mail: panda@cse.ohio-state.edu
http://www.cse.ohio-state.edu/~panda
Outline

• Exascale Computing and Hybrid Programming Model
• Challenges in unifying UPC and MPI
• Solutions and Experimental Results
• Challenges in unifying MPI and OpenSHMEM
• Solutions and Experimental Results
• Conclusions
High-End Computing (HEC): PetaFlop to ExaFlop

Projected Performance Development

Expected to have an ExaFlop system in 2019-2020!
Exaflop Computing

• Exaflop = $10^{18}$ floating point operations per second
• Represents a factor of 100-1000x from current state of the art
• Goal – Reach Exaflop levels by 2019-2020
• Exaflop computing is expected to spur research into high performance technologies
• Discover new technologies to enable next generation of science
# Exascale System Targets

<table>
<thead>
<tr>
<th>Systems</th>
<th>2010</th>
<th>2018</th>
<th>Difference Today &amp; 2018</th>
</tr>
</thead>
<tbody>
<tr>
<td>System peak</td>
<td>2 PFlop/s</td>
<td>1 EFlop/s</td>
<td>O(1,000)</td>
</tr>
<tr>
<td>Power</td>
<td>6 MW</td>
<td>~20 MW (goal)</td>
<td></td>
</tr>
<tr>
<td>System memory</td>
<td>0.3 PB</td>
<td>32 – 64 PB</td>
<td>O(100)</td>
</tr>
<tr>
<td>Node performance</td>
<td>125 GF</td>
<td>1.2 or 15 TF</td>
<td>O(10) – O(100)</td>
</tr>
<tr>
<td>Node memory BW</td>
<td>25 GB/s</td>
<td>2 – 4 TB/s</td>
<td>O(100)</td>
</tr>
<tr>
<td>Node concurrency</td>
<td>12</td>
<td>O(1k) or O(10k)</td>
<td>O(100) – O(1,000)</td>
</tr>
<tr>
<td>Total node interconnect BW</td>
<td>3.5 GB/s</td>
<td>200 – 400 GB/s (1:4 or 1:8 from memory BW)</td>
<td>O(100)</td>
</tr>
<tr>
<td>System size (nodes)</td>
<td>18,700</td>
<td>O(100,000) or O(1M)</td>
<td>O(10) – O(100)</td>
</tr>
<tr>
<td>Total concurrency</td>
<td>225,000</td>
<td>O(billion) + [O(10) to O(100) for latency hiding]</td>
<td>O(10,000)</td>
</tr>
<tr>
<td>Storage capacity</td>
<td>15 PB</td>
<td>500 – 1000 PB (&gt;10x system memory is min)</td>
<td>O(10) – O(100)</td>
</tr>
<tr>
<td>IO Rates</td>
<td>0.2 TB</td>
<td>60 TB/s</td>
<td>O(100)</td>
</tr>
<tr>
<td>MTTI</td>
<td>Days</td>
<td>O(1 day)</td>
<td>-O(10)</td>
</tr>
</tbody>
</table>

 Courtesy: DOE Exascale Study and Prof. Jack Dongarra
Global view improves programmer productivity

Idea is to decouple data movement with process synchronization

Processes should have asynchronous access to globally distributed data

Well suited for irregular applications and kernels that require dynamic access to different data
Different Approaches for Supporting PGAS Models

- **Library-based**
  - Global Arrays
  - OpenSHMEM

- **Compiler-based**
  - Unified Parallel C (UPC)
  - Co-Array Fortran (CAF)

- **HPCS Language-based**
  - X10
  - Chapel
  - Fortress
Outline

- Exascale Computing and Hybrid Programming Model
- Challenges in unifying UPC and MPI
  - Solutions and Experimental Results
- Challenges in unifying MPI and OpenSHMEM
  - Solutions and Experimental Results
- Conclusions
**Issues and Problems**

- Parts of big applications and third party libraries use MPI
- Parallel Math and Physics libraries have very high investment, *cannot* re-write them!
- Separate runtimes for MPI and UPC/OpenSHMEM?
  - Requires more network resources
  - Must ensure progress of both MPI and UPC/OpenSHMEM runtimes
  - May even lead to deadlock!
  - Issues with performance and scalability
  - Don’t interoperate very well
- No unified runtime to support both MPI and UPC/OpenSHMEM over OFED with best performance and scalability
  - Current performance comparison between MPI and UPC/OpenSHMEM is misleading
- No unified runtime to design hybrid programs (MPI+UPC or MPI+OpenSHMEM) on emerging multi-core environments
Various ways to use UPC and MPI and Limitations

Not as scalable as MVAPICH / MVAPICH2 on OFED

UPC semantics Mapped onto MPI
Bad performance!!
What is the way forward?

• Can we place UPC on top of MPI?
  – Active messages (AM) not part of MPI; critical to UPC
  – UPC is lighter-weight, so putting on top of MPI loses performance
  – Other model mismatches (some may be solved by MPI-3)

• *Path forward: unify runtimes, not programming models*
Problem Statement

• Can we design a communication library for UPC?
  – Scalable on large InfiniBand clusters with RDMA
  – Provides equal or better performance than existing runtime

• Can this library support both MPI and UPC?
  – Individually, both with great performance
  – Simultaneously, with great performance and less memory
Benefits

• Allow scientists to develop applications in the following modes
  • MPI only
  • PGAS (UPC) only
  • Hybrid (MPI and UPC)

• Allow scientists to evaluate the impact of programming models on applications on next generation systems in a fair manner
Outline

• Exascale Computing and Hybrid Programming Model
• Challenges in unifying UPC and MPI
  • Solutions and Experimental Results
  • Challenges in unifying MPI and OpenSHMEM
  • Solutions and Experimental Results
• Conclusions
Unifying UPC and MPI Runtimes: Experience with MVAPICH2

- Currently UPC and MPI do not share runtimes
  - Duplication of lower level communication mechanisms
  - GASNet unable to leverage advanced buffering mechanisms developed for MVAPICH2
- Our novel approach is to enable a truly unified communication library
New Configuration for UPC and MPI
**UPC Micro-benchmark Performance**

- BUPC micro-benchmarks from latest release 2.10.2
- UPC performance is identical with both native IBV layer and new UCR layer
- Performance of GASNet-MPI conduit is not very good
  - Mismatch of MPI specification and Active messages
- GASNet-UCR is more scalable compared native IBV conduit

Evaluation using UPC NAS Benchmarks

- GASNet-UCR performs equal or better than GASNet-IBV
- 10% improvement for CG (B, 128)
- 23% improvement for MG (B, 128)
Evaluation of Hybrid MPI+UPC NAS-FT

- Modified NAS FT UPC all-to-all pattern using MPI_Alltoall
- Truly hybrid program
- 34% improvement for FT (C, 128)
Graph500 Results with new UPC Queue Design

- Workload – Scale:24, Edge Factor:16 (16 million vertices, 256 million edges)
- 44% Improvement over base version for 512 UPC-Threads
- 30% Improvement over base version for 1024 UPC-Threads

Outline

• Exascale Computing and Hybrid Programming Model
• Challenges in unifying UPC and MPI
• Solutions and Experimental Results
• Challenges in unifying MPI and OpenSHMEM
• Solutions and Experimental Results
• Conclusions
Scalable OpenSHMEM and Hybrid (MPI and OpenSHMEM) designs

- Based on OpenSHMEM Reference Implementation
  - Provides a design over GASNet
  - Does not take advantage of all OFED features

- Design scalable and High-Performance OpenSHMEM over OFED

- Designing a Hybrid MPI +OpenSHMEM Model
  - Current Model – Separate Runtimes for OpenSHMEM and MPI
    - Possible deadlock if both runtimes are not progressed
    - Consumes more network resource
  - Our Approach – Single Runtime for MPI and OpenSHMEM
Micro-Benchmark Performance (OpenSHMEM)

Atomic Operations

- fadd
- finc
- cswap
- swap
- add
- inc

Time (us)

OpenSHMEM-GASNet
OpenSHMEM-UCR

41% 16%

OpenSHMEM-GASNet
OpenSHMEM-UCR

Broadcast (256 bytes)

Time (ms)

No. of Processes

32 64 128 256

Collect (256 bytes)

Time (ms)

No. of Processes

32 64 128 256

Reduce (256 bytes)

Time (ms)

No. of Processes

32 64 128 256

41% 36% 35%
Performance of OpenSHMEM Applications

- 2D FFT with 8K input matrix
  - 16% improved performance for 512 processes
- 2D Heat Transfer Modeling
  - 45% improved performance for 512 processes
- Performance Improvement because of high performance runtime
Performance of Hybrid (OpenSHMEM+MPI) Applications

- Improved Performance for Hybrid Applications
  - 34% improvement for 2D Heat Transfer Modeling with 512 processes
  - 45% improvement for Graph500 with 256 processes
- Our approach with single Runtime consumes 27% lesser network resources
Conclusions

• Hybrid programming models are critical for Exascale systems

• Unified Communication Runtime (UCR)
  • Supports MPI+UPC and MPI+OpenSHMEM simultaneously on OFED using RDMA features

• Promising:
  • MPI communication not harmed
  • \{UPC, OpenSHMEM\} communication performance and scalability are improved

• Allows to solve problems using multiple programming modes
  • MPI only
  • PGAS (UPC) only
  • PGAS (OpenSHMEM)
  • Hybrid (MPI and UPC)
  • Hybrid (MPI and OpenSHMEM)

• Suitable candidate for Exascale Computing
Web Pointers

http://www.cse.ohio-state.edu/~panda
http://nowlab.cse.ohio-state.edu

MVAPICH Web Page
http://mvapich.cse.ohio-state.edu

panda@cse ohio state edu