Reduction Operations on Modern Supercomputers: Challenges and Solutions

Mohammadreza Bayatpour, Jahanzeb Maqbool Hashmi, Sourav Chakraborty, Hari Subramoni, Dhabaleswar K. Panda

THE OHIO STATE UNIVERSITY

Overview

Importance of Reduction Operations:

- One of the most popular MPI collectives
- Widely used in Deep Learning frameworks and Scientific applications
- Extensive usage of compute resources as well as network

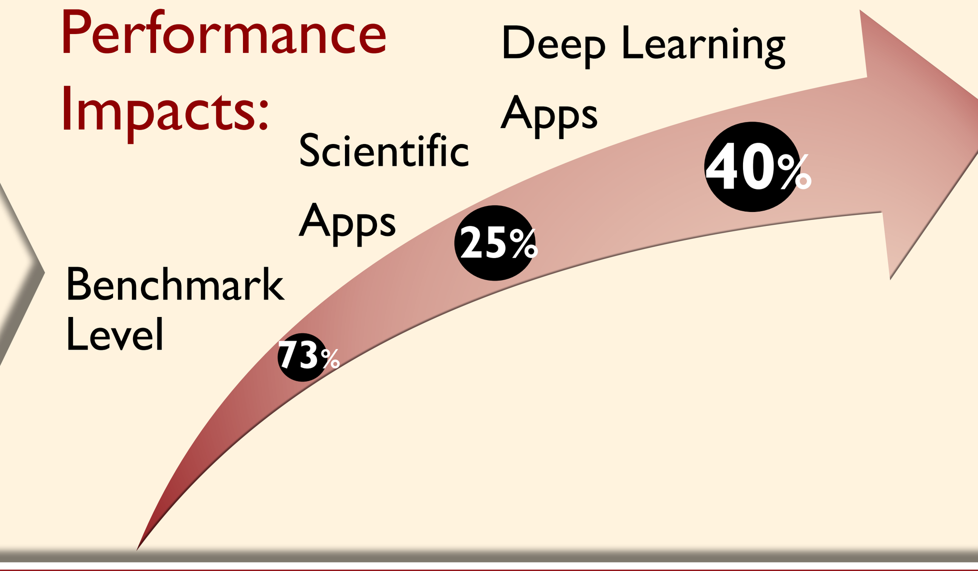
Research Challenges:

- Efficient usage of network offload mechanisms and high-throughput network
- Enhanced usage of one-sided semantics and cache locality
- Efficient pipeline and overlap across various design phases
- Dynamic and adaptive communication

Proposed Solutions:

- Enhanced SHArP network offload
  - Target: Small Messages
- Data Partitioning-based Multi-leader design
  - Target: Medium Messages
- XPMEM/SHMEM-based Scalable and adaptive design
  - Target: Large Messages

Performance Impacts:



Challenges

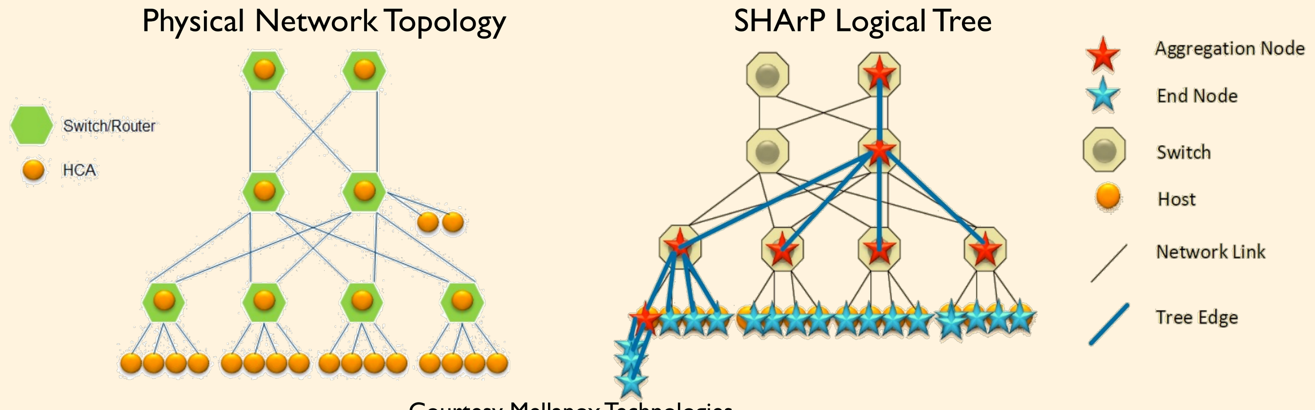
Small Messages

Approaches

1. Onloading approach: CPU-assisted approach
2. Offloading approach: using HCA (CORE-Direct) or Switch (SHArP)

Scalable Hierarchical Aggregation Protocol (SHArP)

- Manipulation of data while it is being transferred in the switch network



Courtesy Mellanox Technologies

Challenges

- Current designs are not NUMA-aware
- Limited performance due to extra cross socket transfers
- Low performance for medium and large message ranges

Medium Messages

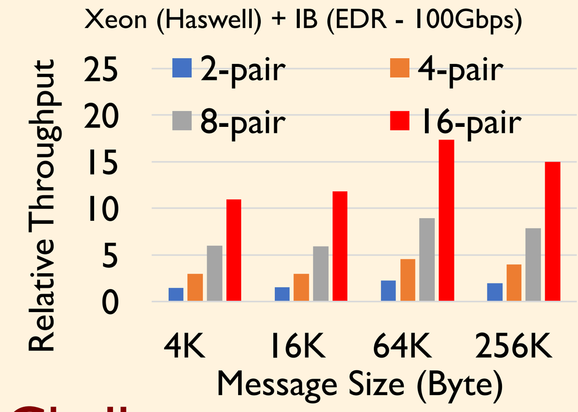
Approaches

1. Topology-aware (hierarchal): Two-level designs (intra-node reduce + inter-node Allreduce)
2. Flat designs: Tree-based designs

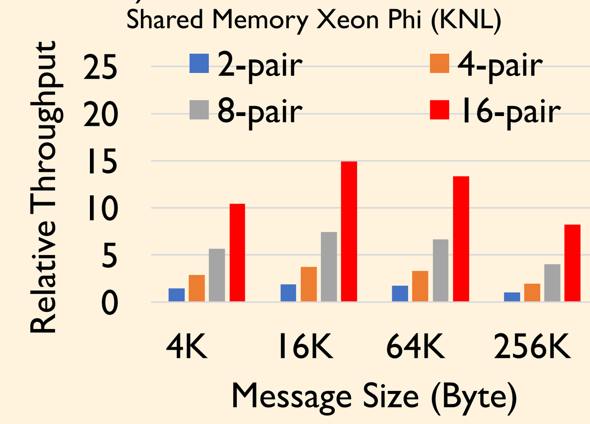
Communication Characteristics of Modern Architectures

- Supports many concurrent intra-node as well as inter-node communications (similar to Omni-Path)

Xeon (Haswell) + IB (EDR - 100Gbps)



Shared Memory Xeon Phi (KNL)



Challenges

- Do not take advantage of high concurrency in new architectures (Hierarchical designs)
- Too many inter-node communication and deep hierarchy (Tree-based designs)

Large Messages

Approaches

1. Intra-node zero copy mechanism
2. Inter-node one-sided communications
3. Inter-node pipelining with intra-node operations
4. Pipelined inter-node Allreduce
5. Communication Adaptive

Various Designs:	Optimization Methods					
	Applicability	1	2	3	4	5
Baidu-Allreduce (c)	GPU	✗	✗	✓	✓	✗
Linear Pipelining (d)	GPU	✗	✗	✓	✓	✗
Reduce-scatter-Allgather	CPU/GPU	✗	✗	✗	✗	✗
Segmented Ring (e)	GPU/CPU	✗	✗	✓	✓	✗
XPMEM-based Reduction (f)	CPU	✓	✗	✗	✗	✗
Proposed "SALaR"	CPU	✓	✓	✓	✓	✓

Challenges

- Efficient pipeline of various steps and usage of XPMEM/SHMEM
- Efficient utilization of compute resources in all processes
- Orchestrating the data transfers to effectively utilize the network bandwidth without oversubscribing a particular link

Proposed Designs

Naive SHArP Design

- SHArP only used in inter-node reduction operation
- Step 1: Intra-node reduction by one process in each node
- Step 2: Then Inter-node Allreduce using SHArP
- Step 3: Broadcast the final results from node-leader to other processes

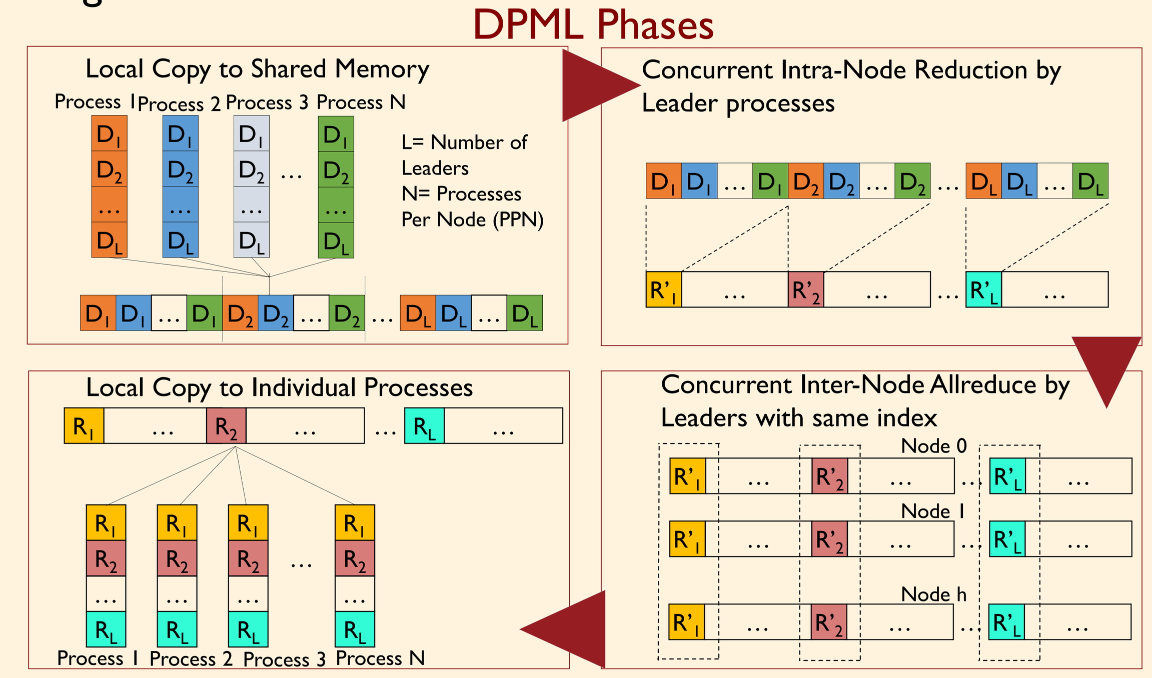
NUMA-Aware SHArP Design (a)

- Mixture of the CPU-assisted designs with Offloaded approaches
- Topology-aware (hierarchal): Two-level designs
- Introducing socket-level leader process to to limit the QPI transfers
- Allowing the leader process in each socket to use SHArP
- Using CPU for intra-socket reduction operations

Data Partitioning based Multi-Leader (DPML) Design (a)

- Having shallow hierarchies with small depth and large number of children per parent
- Taking advantage of high-throughput of concurrent medium messages

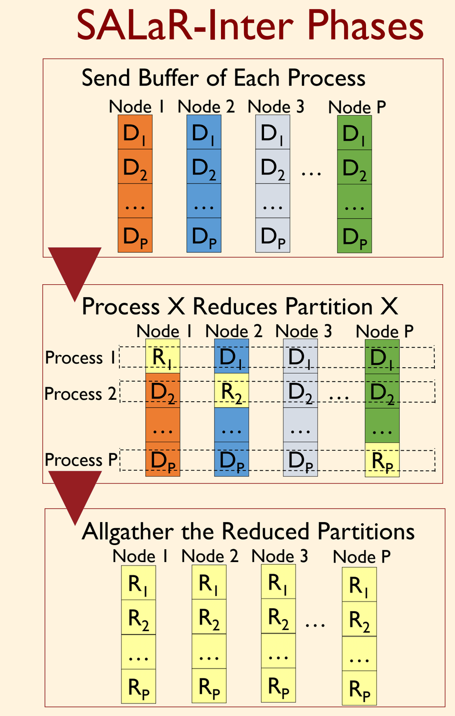
DPML Phases



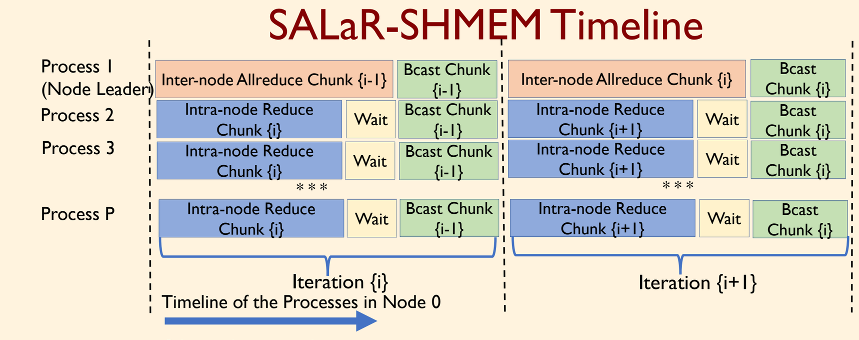
Scalable and Adaptive Designs for Large Messages Reduction Collectives (SALaR) (b)

1. SALaR-SHMEM/XPMEM: A pipelined Allreduce design which uses XPMEM/SHMEM for intra-node reduction and SALaR-Inter for inter-node reduction. Intra-node operation is overlapped with inter-node operation.
2. SALaR-Inter: An efficient one-sided-based Inter-node Allreduce

SALaR-Inter Phases



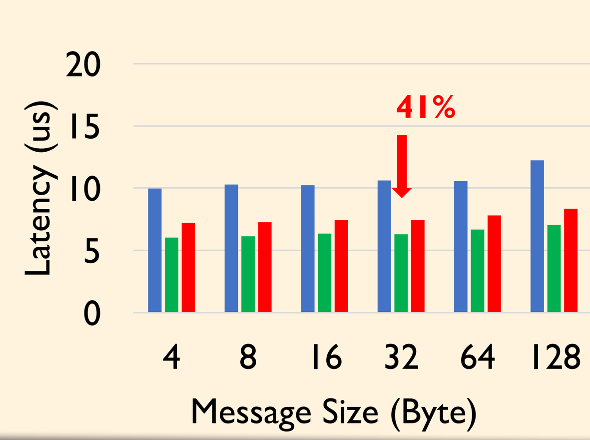
SALaR-SHMEM Timeline



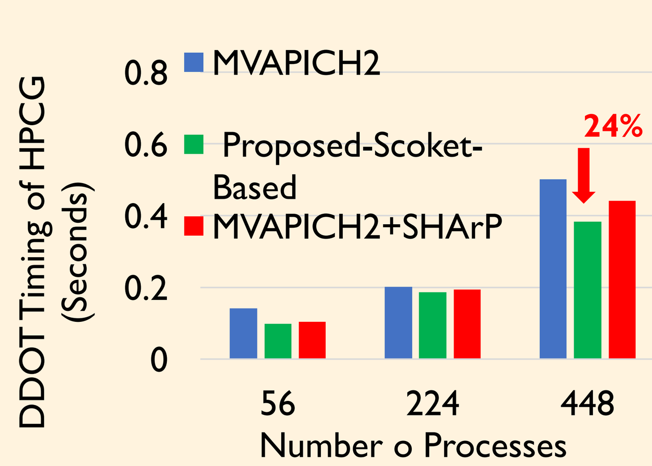
Impact

Performance of SHArP Designs on XEON+IB

MPI_Allreduce (448 Processes, 16 Nodes, 28 PPN)

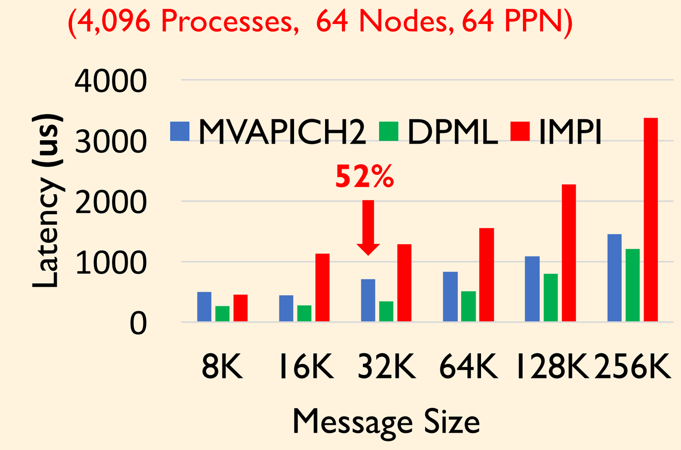


HPCG (28 PPN)

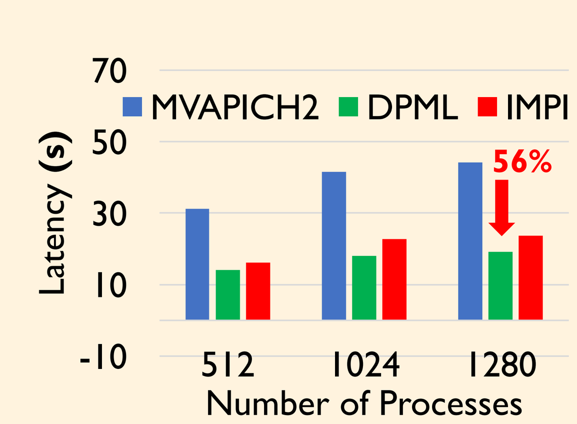


Performance of DPML Designs on KNL+Omni-Path

MPI_Allreduce (4,096 Processes, 64 Nodes, 64 PPN)

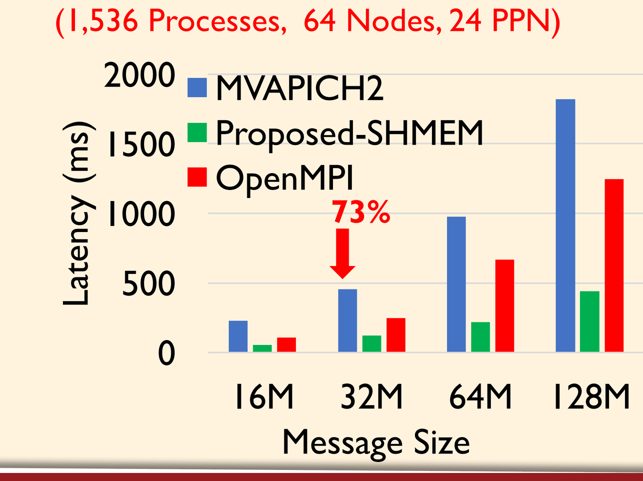


MiniAMR (32 PPN)

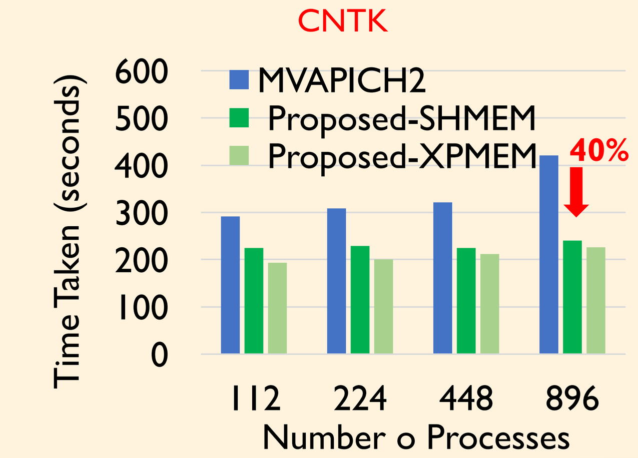


Performance of SALaR Designs on XEON+IB

MPI_Allreduce (1,536 Processes, 64 Nodes, 24 PPN)



AlexNet Neural Network Training using CNTK



References:
(a) Scalable Reduction Collectives with Data Partitioning-based Multi-Leader Design, **Bayatpour** et al, Supercomuting'17, Denver, Co.
(b) SALaR: Scalable and Adaptive Designs for Large Message Reduction Collectives, Bayatpour et al, IEEE Cluster'18, Belfast. UK
(c) Baidu Allreduce Design: https://github.com/baidu- research/baidu-allreduce
(d) Efficient communications in training large scale neural networks, Zhao et al, Thematic Workshops ACMMM2017
(e) Bandwidth optimal all-reduce algorithms for clusters of workstations, Patarasuk et al, Journal of Parallel and Distributed Comp '09
(f) Designing Efficient Shared Address Space Reduction Collectives for Multi-/Many-cores, Hashmi et al, IPDPS '17