Design Alternatives for Implementing Fence Synchronization in MPI-2 One-Sided Communication for InfiniBand Clusters

G.Santhanaraman, T. Gangadharappa, S.Narravula,
A.Mamidala and D.K.Panda

Presented by: Miao Luo

National Center for Supercomputing Applications
Dept of Computer Science and Engineering, The Ohio State University

Introduction

- High-end Computing (HEC) Systems (approaching petascale capability)
  - Systems with few thousands/tens/hundreds of thousands of cores
  - Meet the requirements of grand challenge problems
- Greater emphasis on programming models
  - One sided communication is getting popular
    - Minimize the need to synchronization
  - Ability to overlap computation and communication
- Scalable application communication patterns
  - Clique-based communication
    - Nearest neighbor: Ocean/Climate modeling, PDE solvers
    - Cartesian grids: 3DFFT

Introduction:

HPC Clusters

- HPC has been the key driving force
  - Provides immense computing power by increasing the scale of parallel machines
- Approaching petascale capabilities
  - Increased Node performance
  - Faster/Larger Memory
  - Hundreds of thousands of cores
- Commodity clusters with Modern Interconnects (InfiniBand, Myrinet 10GigE etc)

Introduction:

Message Passing Interface (MPI)

- MPI - Dominant programming model
- Very Portable
  - Available on all High end systems
- Two sided message passing
  - Requires a handshake between the sender and receiver
  - Matching sends and receives
- One sided programming models becoming popular
  - MPI also provides one-sided communication semantics

Introduction:

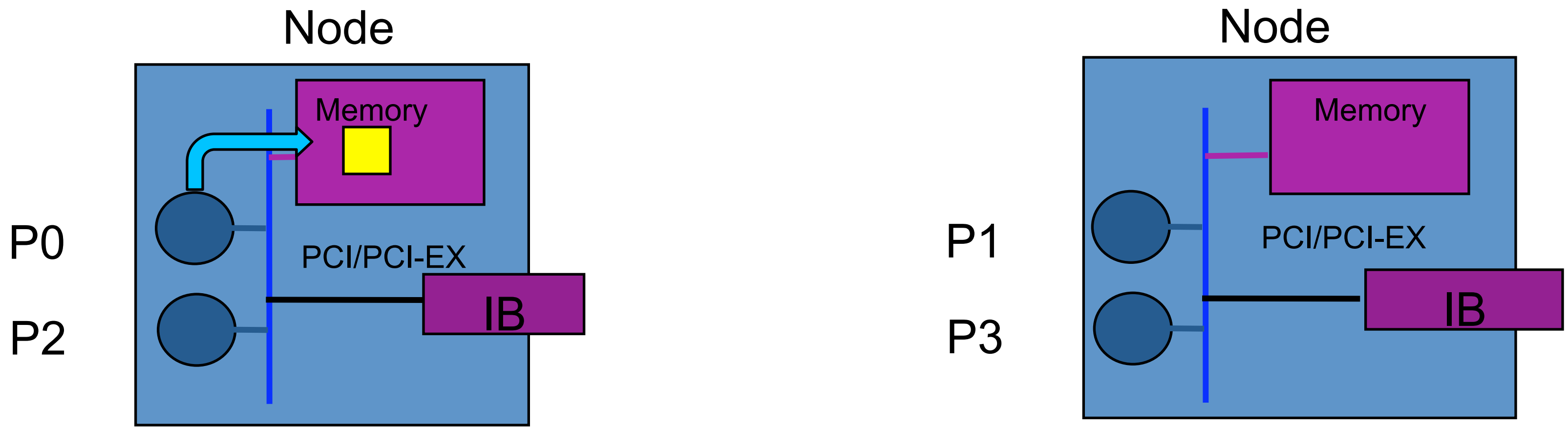
One-sided Communication

- P0 reads/writes directly into the address space of P1

- Only one processor (P0) involved in the communication

- MPI-2 standard (extension to MPI-1)
  One Sided Communication or
  Remote memory Access (RMA)

MPI-3 standard coming up...



MPI-2 One-sided Communication

- Sender (origin)
  ca
  n
  access the receiver (target) remote address space (window) directly

- Decouples data transfer and synchronization operations

- Communication operations
  - MPI_Put, MPI_Get, MPI_Accumulate
  - Contiguous and Non-contiguous operations

- Synchronization Modes
  - Active synchronization
    - Post/start Wait/Complete
    - **Fence (collective)**
  - Passive synchronization
    - Lock/unlock

Introduction:

Fence Synchronization

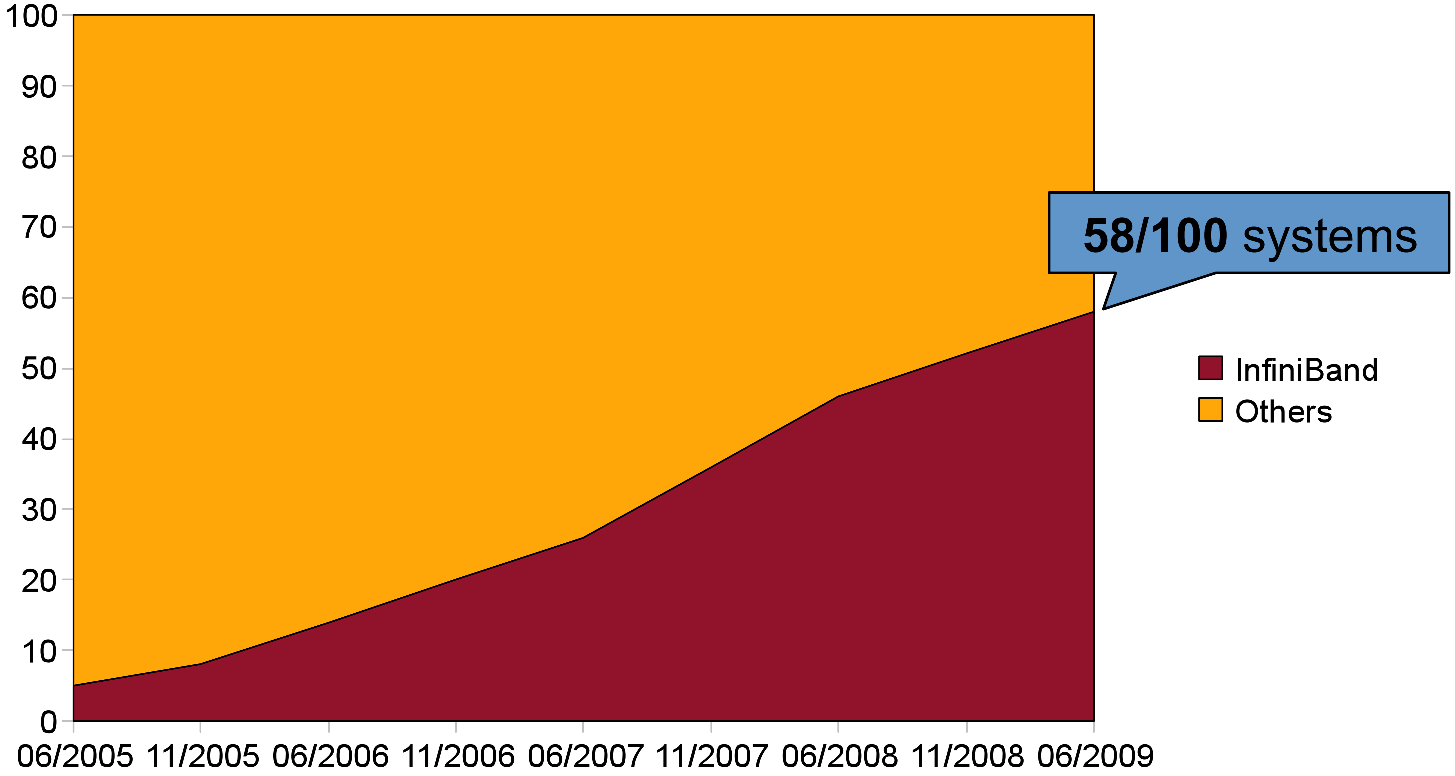
	PROCESS: 0	PROCESS : 1	PROCESS: 2
START: epoch 0	Fence	Fence	Fence

	Put(2)
	Get(1)
		Put(2)	Put(0)

END : epoch 0 START: epoch 1	Fence	Fence	Fence
	Put(1)	Put(2)	Put(1)
END: epoch 1	Fence	Fence	Fence

Introduction:

Top 100 Interconnect Share



58/100 systems

In top systems, the use of InfiniBand has grown significantly.

Over 50% of the top 100 systems in the Top500 use InfiniBand

Introduction:

InfiniBand Overview

- The InfiniBand Architecture (IBA):
  Open standard for high speed interconnect

- IBA supports send/recv and RDMA semantics
  - Can provide good hardware support for RMA/one-sided communication model

- Very good performance with many features
  - Minimum latency ~1usecs, peak bandwidth ~2500MB/s
  - RDMA Read, RDMA Write (matches well with one-sided get/put semantics)
  - *RDMA Write with Immediate (explored in this work)*

- Several High End Computing systems use InfiniBand
  examples: Ranger at TACC (62976 cores), Chinook at PNNL (18176 cores)

Presentation Layout

- Introduction
- ***Problem Statement***
- Design Alternatives
- Experimental Evaluation
- Conclusions and Future Work

Problem Statement

- How can we explore the design space for implementing fence synchronization on modern Interconnects?

- Can we design a novel fence synchronization mechanism that leverages InfiniBand's RDMA Write with immediate primitives?
  - Reduced synchronization overhead and network traffic
  - Provide increased scope for overlap

Presentation Layout

- Introduction
- Problem Statement
- ***Design Alternatives***
- Experimental Evaluation
- Conclusions and Future Work

Design Space

- Deferred Approach
  - All operations and synchronizations deferred to subsequent fence
  - Use two-sided operations
  - Certain optimizations possible to reduce latency of ops and overhead of sync
  - Capability for overlap is lost
- Immediate Approach
  - Sync and communication ops happen as they are issued
  - Use RDMA for communication ops
  - Can achieve good overlap of computation and communication
  - How can we handle remote completions??
- Characterize the performance
  - Overlap capability
  - Synchronization overhead

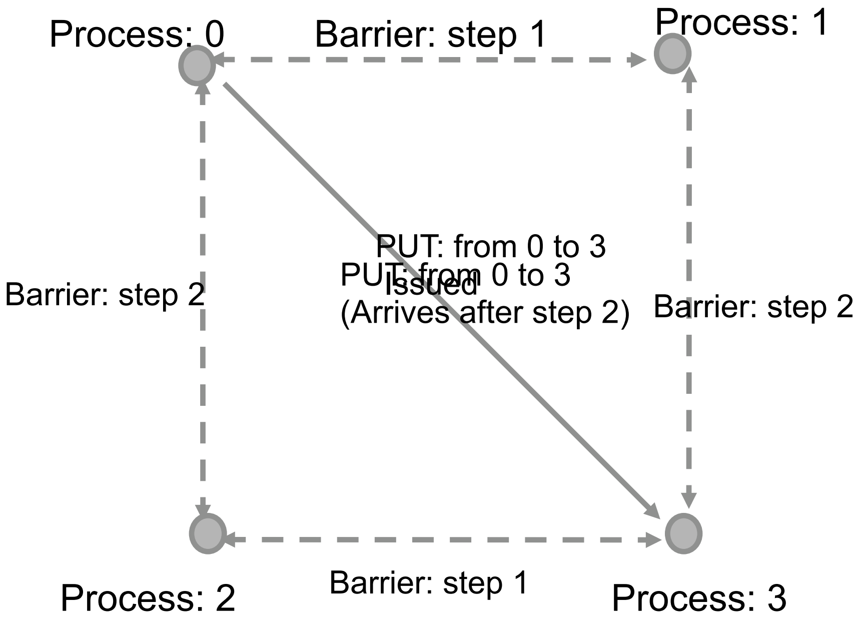
Fence Designs

- Deferred approach (**Fence-2S**)
  - Two Sided Based Approach
  - First fence does nothing
  - All one-sided operations queued locally
  - The second fence goes through the queue, issues operations, and handles completion
  - The last message in the epoch can signal a completion

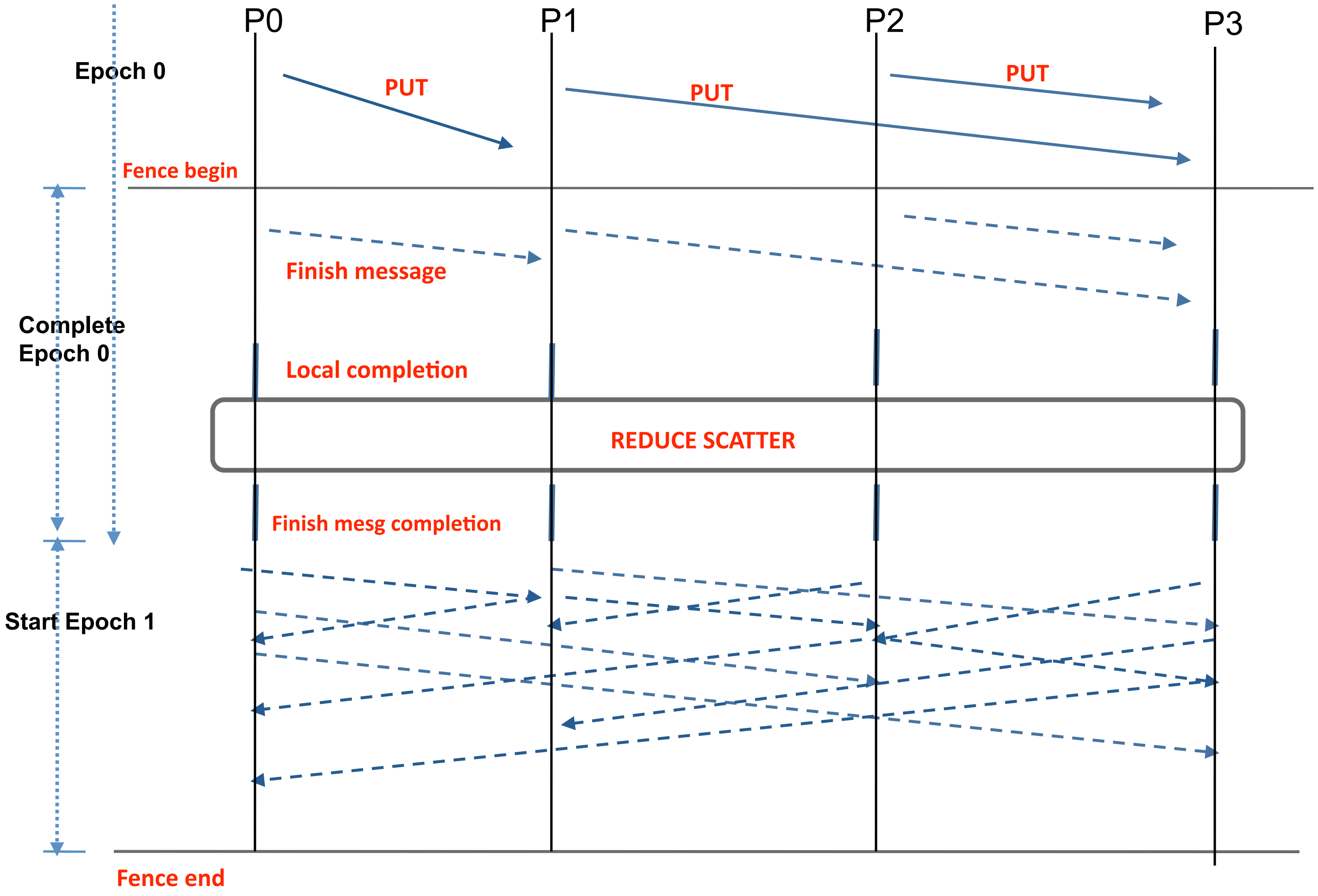
- Optimizations (combining of put and the ensuing synchronization) -> reduced synchronization overhead
- Cons : No scope for providing overlap

Fence Designs

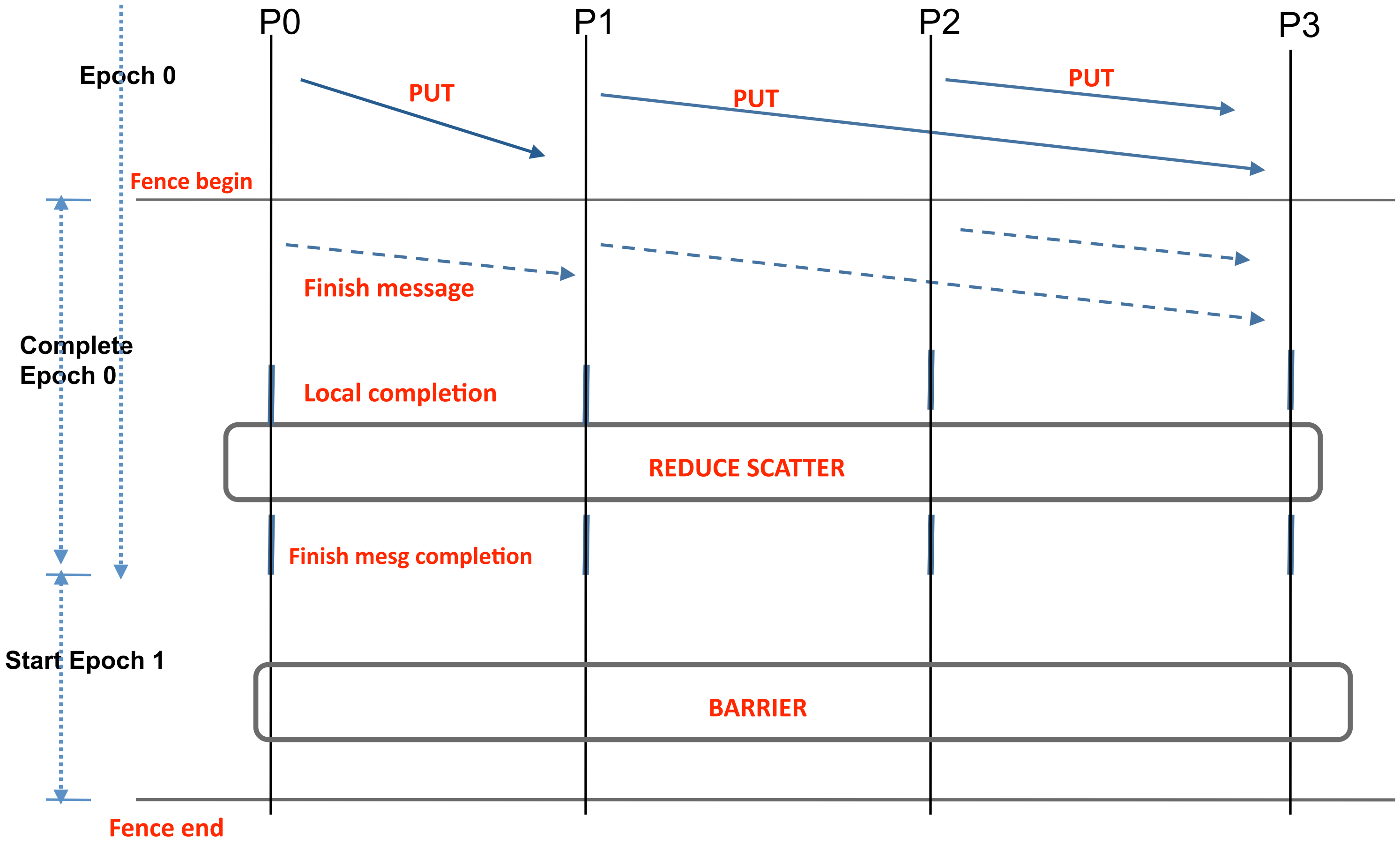
- Immediate Approach
  - Issue a completion message on all the channels
  - Issue a Barrier after the operations?



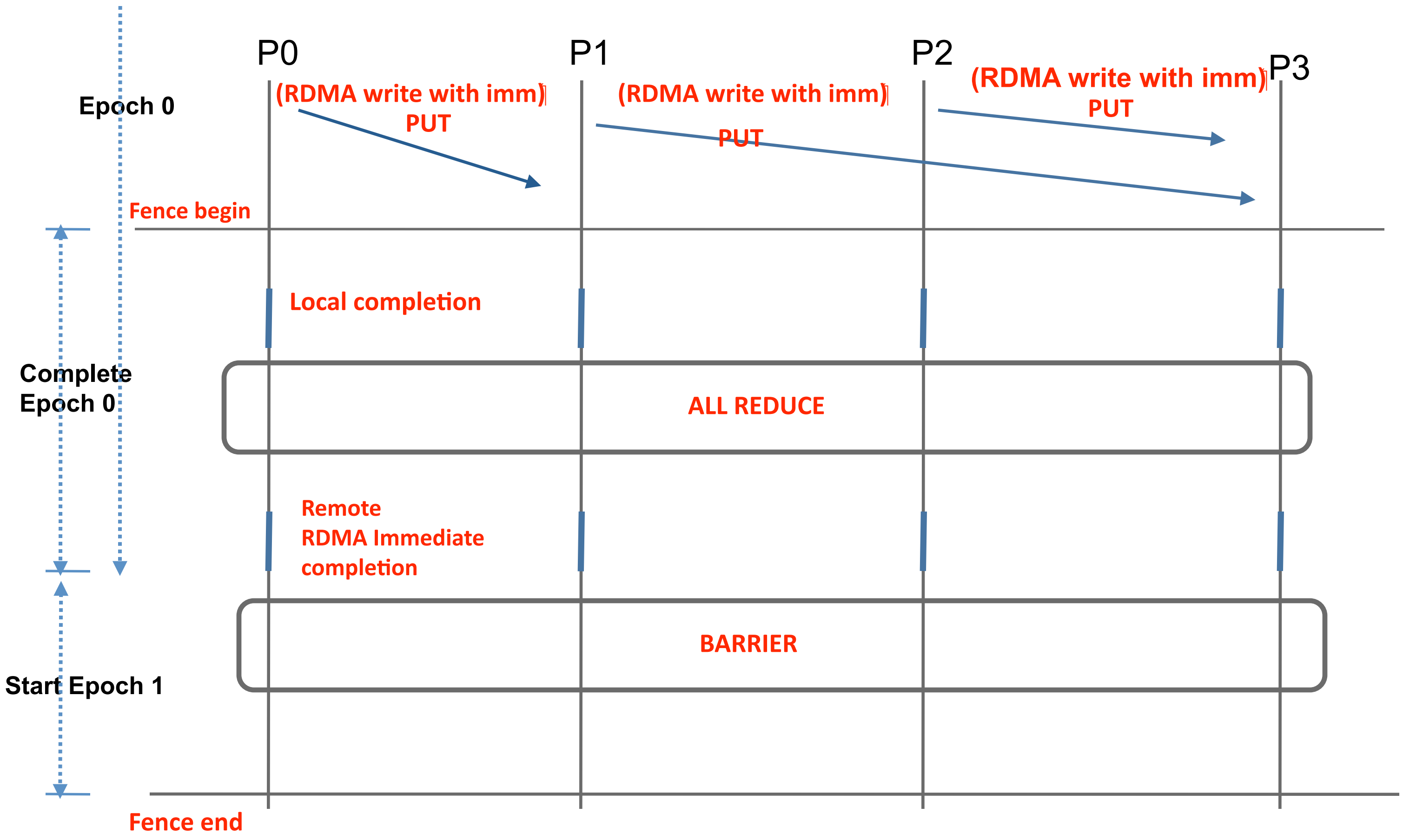
Fence-Imm Naive Design (Fence-1S)



Fence-Imm Opt Design (Fence-1S-Barrier)



Novel Fence-RI Design



Presentation Layout

- Introduction
- Problem Statement
- Design Alternatives
- ***Experimental Evaluation***
- Conclusions and Future Work

Experimental Evaluation

Experimental Testbed

- 64 Node Intel Cluster
- 2.33 GHz quad-core processor
- 4GB Main Memory
- RedHat Linux AS4
- Mellanox MT25208 HCAs with PCI Express Interfaces
- Silverstorm 144 port switch
- MVAPICH2 Software Stack

Experiments Conducted

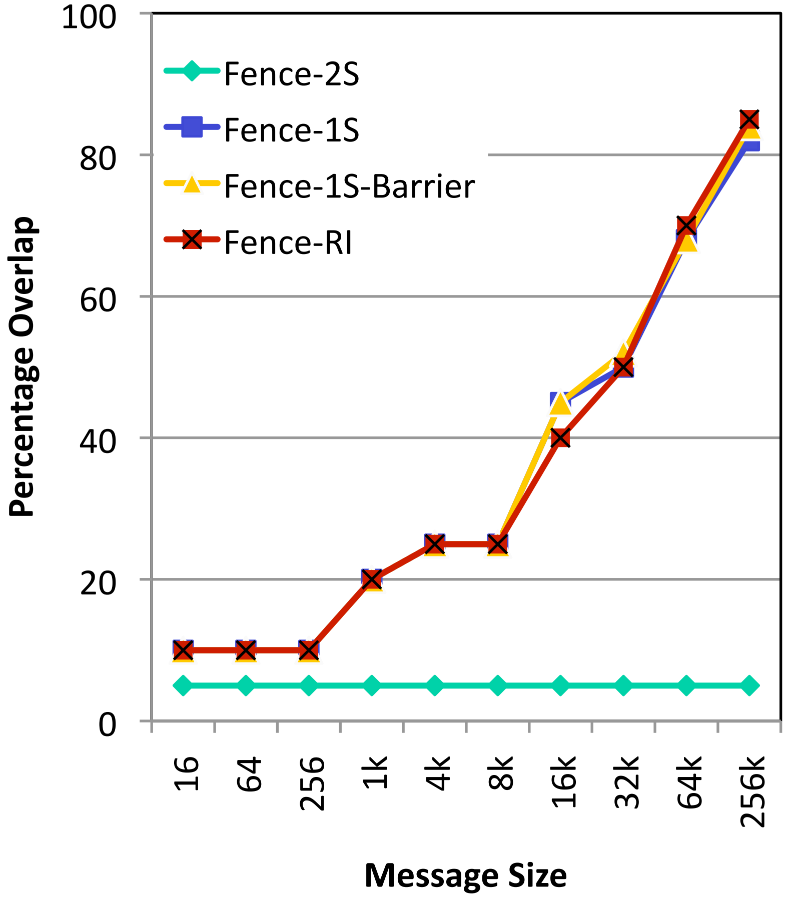
- Overlap Measurements
- Fence Synchronization Microbenchmarks
- Halo Exchange Communication Pattern

MVAPICH/MVAPICH2 Software Distributions

- High Performance MPI Library for InfiniBand and iWARP Clusters
  - MVAPICH2(MPI-2)
  - Used by more than 975 organizations world-wide
  - Empowering many TOP500 clusters
  - Available with software stacks of many InfiniBand, iWARP and server vendors including Open Fabrics Enterprise Distribution (OFED)

  http://mvapich.cse.ohio-state.edu/

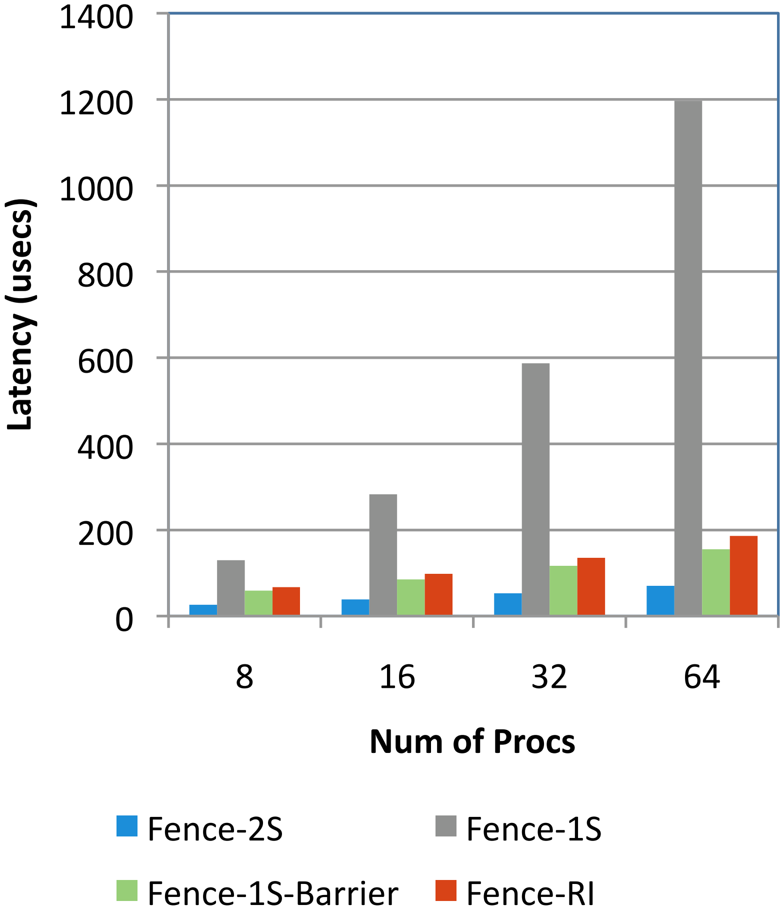
Overlap



- Overlap Metric
  - Increasing amount of computation is inserted between the put and fence sync
  - Percentage overlap is measured as the amount of computation that can be inserted without increasing overall latency

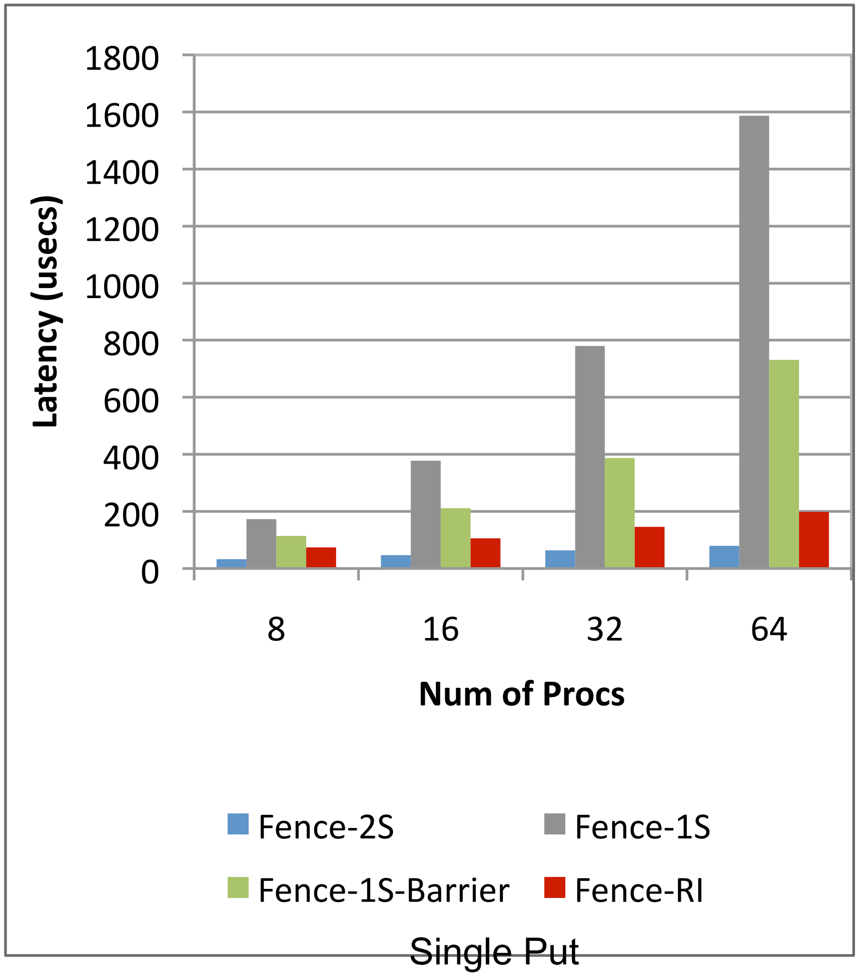
- Two sided implementation (Fence-2S) uses deferred approach
  - No scope for overlap
- The one-sided implementations can achieve overlap

Latency of Fence (Zero-put)



- Performance of fence alone without any one-sided operations
- Overhead of synchronization alone
- Fence-1S performs badly due to all pair-wise sync to indicate start of next epoch
- Fence-2S performs the best since it does not need additional collective to indicate start of an epoch

Latency of Fence with Put Operations

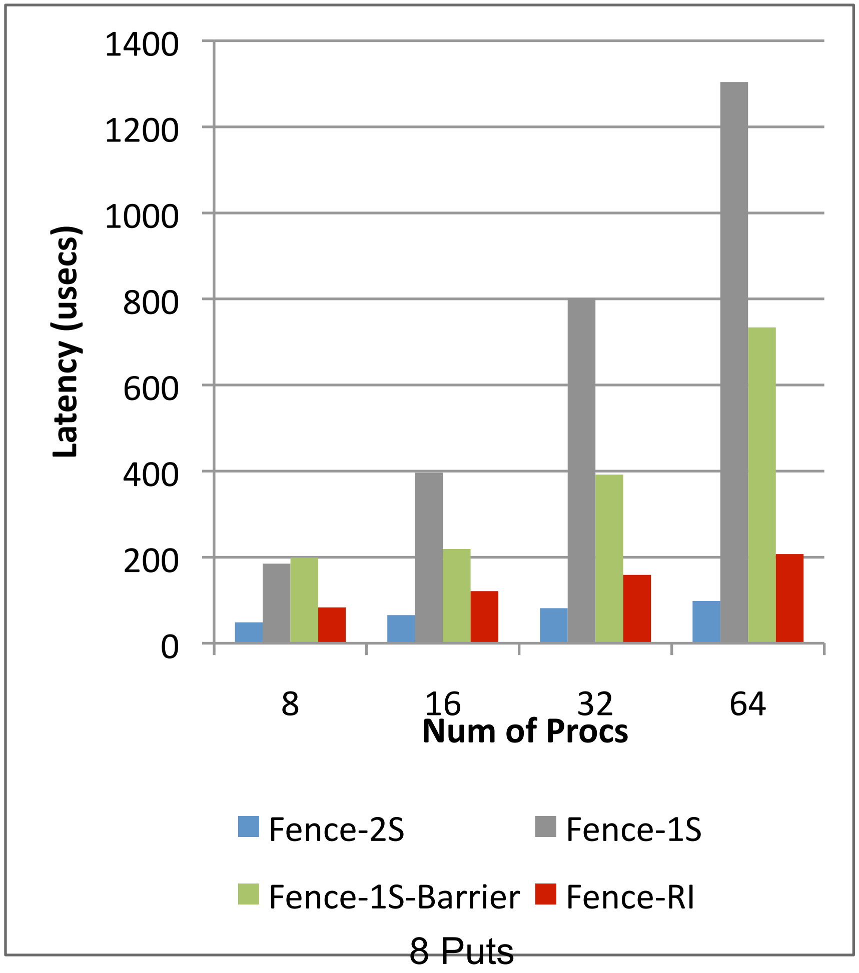


Single Put

- Performance of fence with put operations
  - Measuring synchronization with communication ops
  - A single put is issued by all the processes between two fences

- Fence-1s performs the worst
- Fence-RI performs better than Fence-1S-Barrier

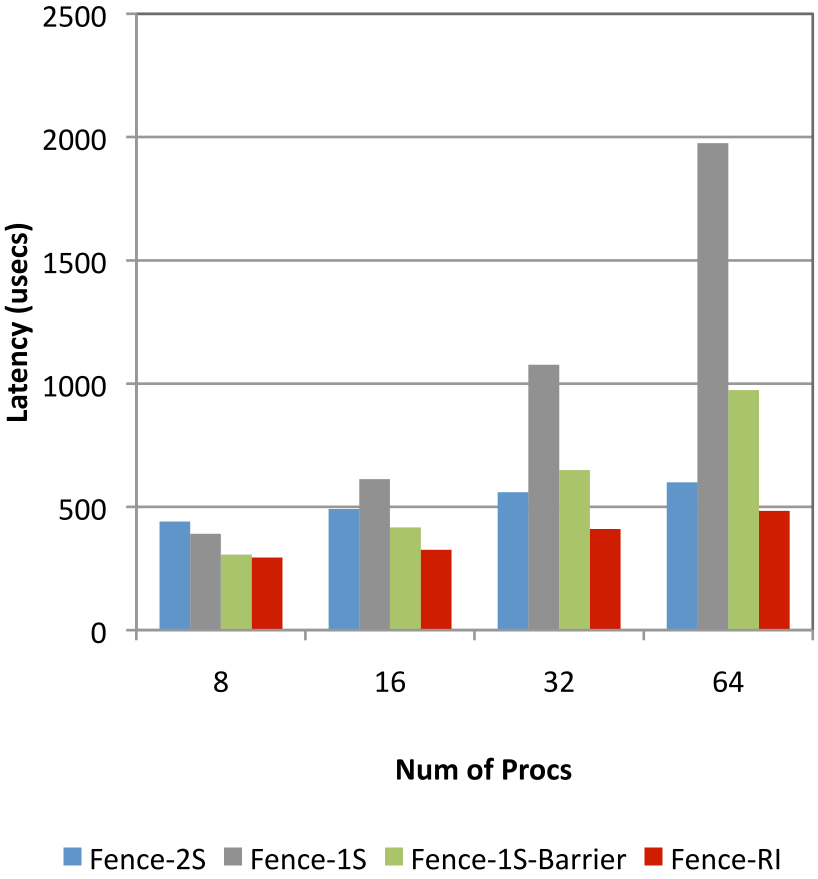
Latency of Fence with Multiple Put Operations

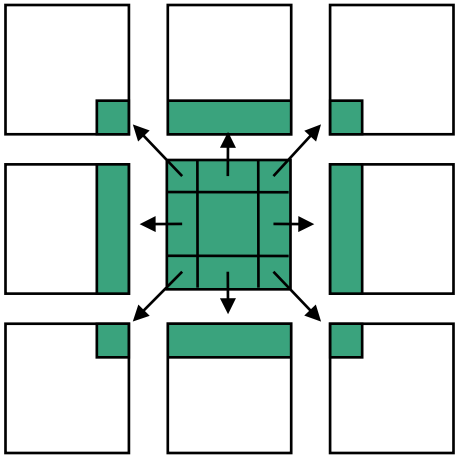


- Performance of fence with multiple put operations
  - Each process issues puts to 8 neighbors

- Fence-RI performs better than Fence-1S barrier
- Fence-2S still performs the best
  - However poor overlap capability

Halo Communication Pattern





- Mimics halo or Ghost cell update
- The Fence-RI scheme performs the best

Presentation Layout

- Introduction
- Problem Statement
- Design Alternatives
- Experimental Evaluation
- ***Conclusions and Future Work***

Conclusions and Future Work

- Analyzed different design choices for implementing fence synchronizations on modern interconnects
- Proposed a new design using RDMA Write with Imm mechanism
  - handle remote completions
- Significantly improved performance for microbenchmarks and application communication patterns
- Future Work
  - Impact of these designs on real world applications

THANK YOU

Email Contacts

G. Santhanaraman: gopal@ncsa.uiuc.edu

D.K. Panda: panda@cse.ohio-state.edu