High Performance Pipelined Process Migration with RDMA

Xiangyong Ouyang, Raghunath Rajachandrasekar, Xavier Besseron, Dhabaleswar K. (DK) Panda

Department of Computer Science & Engineering

The Ohio State University

Outline

- Introduction and Motivation
- Profiling Process Migration
- Pipelined Process Migration with RDMA
- Performance Evaluation
- Conclusions and Future Work

Motivation

- Computer clusters continue to grow larger
  - Heading towards Multi-PetaFlop and ExaFlop Era
  - Mean-time-between-failures (MTBF) is getting smaller
  - ➢ Fault-Tolerance becomes imperative

- Checkpoint/Restart (C/R) – common approach to Fault Tolerance
  - Checkpoint: save snapshots of all processes (**IO overhead**)
  - Restart: restore, resubmit the job (**IO overhead** + **queue delay**)
- C/R Drawbacks
  - × **Unnecessarily dump all processes ➔ IO bottleneck**
  - × **Resubmit queuing delay**

➢Checkpoint/Restart alone doesn't scale to large systems

Job/Process Migration

- Pro-active Fault Tolerance
  - Only handle processes on failing node
  - Health monitoring mechanisms, failure prediction models

- Five steps
  - (1) Suspend communication channels
  - (2) Write snapshots on source node
  - (3) Transfer process image files (Source=>Target)
  - (4) Read image files on target node
  - (5) Reconnect communication channels

Process Migration Advantages

- Overcomes C/R drawbacks
  - × Unnecessary dump of all processes
  - × Resubmit queuing delay
- Desirable feature for other applications
  - Cluster-wide load balancing
  - Server consolidation
  - Performance isolation

Existing MPI Process Migration

- Available in MVAPICH2 and OpenMPI
- Both suffers low performance
- Cause? Solution?

Problem Statements

- What are the dominant factors of the high cost of process migration?
- How to design an efficient protocol to minimize overhead?
  - How to optimize checkpoint-related I/O path ?
  - How to optimize data transfer path?
  - How to leverage RDMA transport to accelerate data transmission?
- What will be the performance benefits?

Outline

- Introduction and Motivation
- Profiling Process Migration
- Pipelined Process Migration with RDMA
- Performance Evaluation
- Conclusions and Future Work

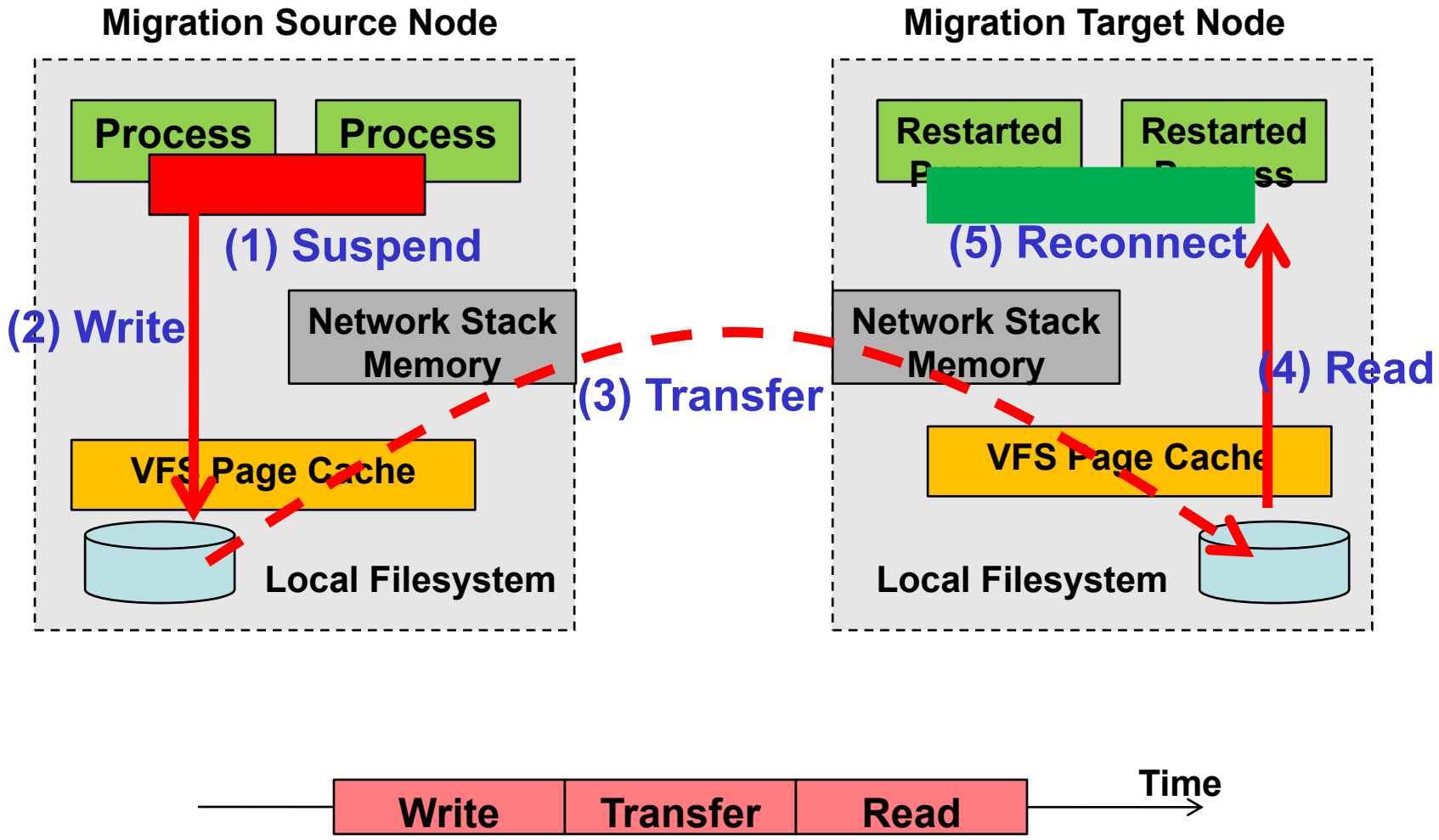
MVAPICH/MVAPICH2 Software

- MVAPICH: MPI over InfiniBand, 10GigE/iWARP and RDMA over Converged Enhanced Ethernet (RoCE)
  - MVAPICH (MPI-1) and MVAPICH2 (MPI-2)
  - Used by more than 1,550 organizations worldwide (in 60 countries)
  - Empowering many TOP500 clusters (11th, 15th …)
  - Available with software stacks of many IB, 10GE/iWARP and RoCE, and server vendors including Open Fabrics Enterprise Distribution (OFED)
  - Available with Redhat and SuSE Distributions
  - http://mvapich.cse.ohio-state.edu/
- Has support for Checkpoint/Restart and Process Migration for the last several years
  - Already used by many organizations

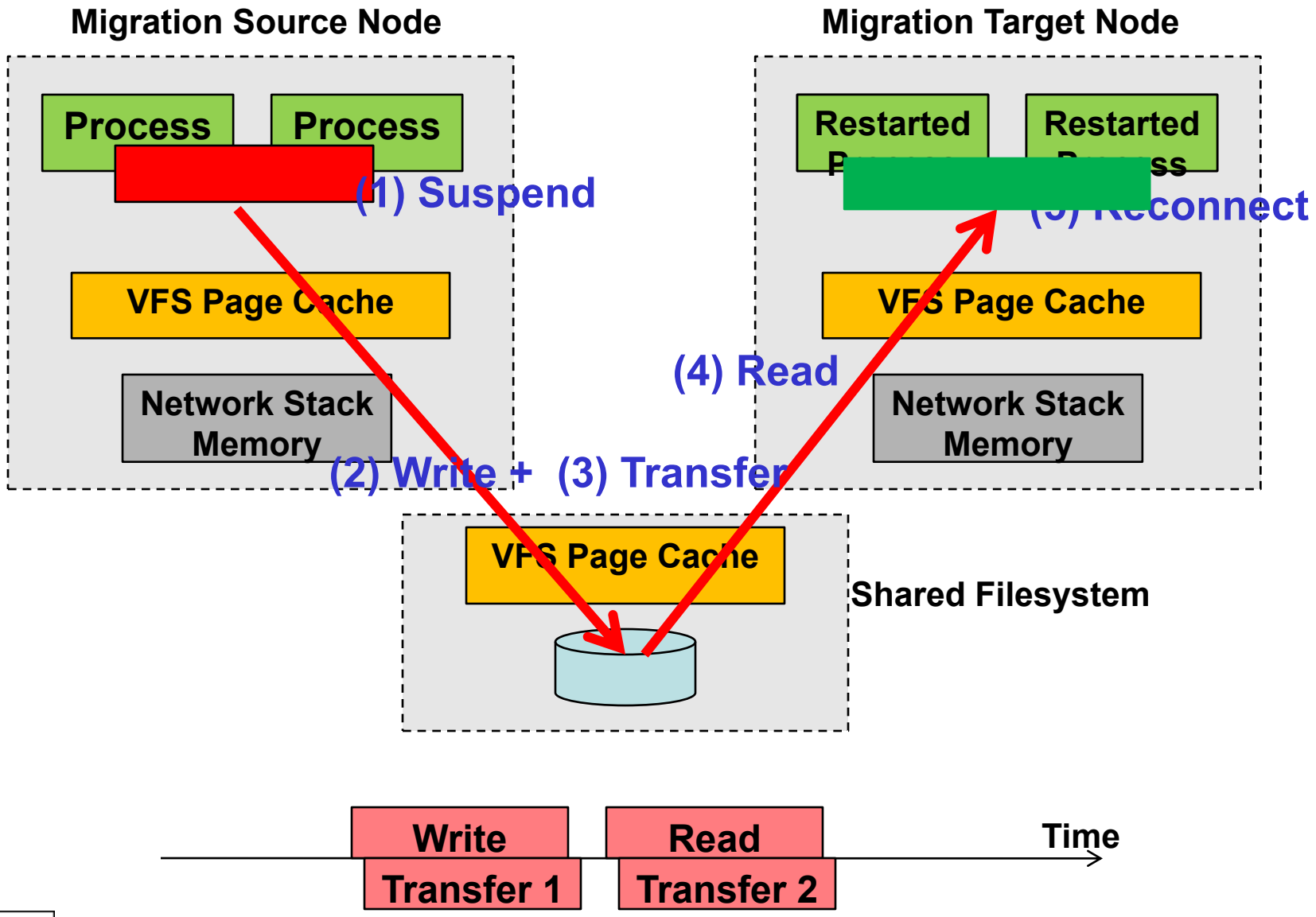
Three Process Migration Approaches

- MVAPICH2 already supports three process migration strategies
  - Local Filesystem-based Migration (*Local*)
  - Shared Filesystem-based Migration (*Shared*)
  - RDMA+Local Filesystem-based Migration (*RDMA+Local)*

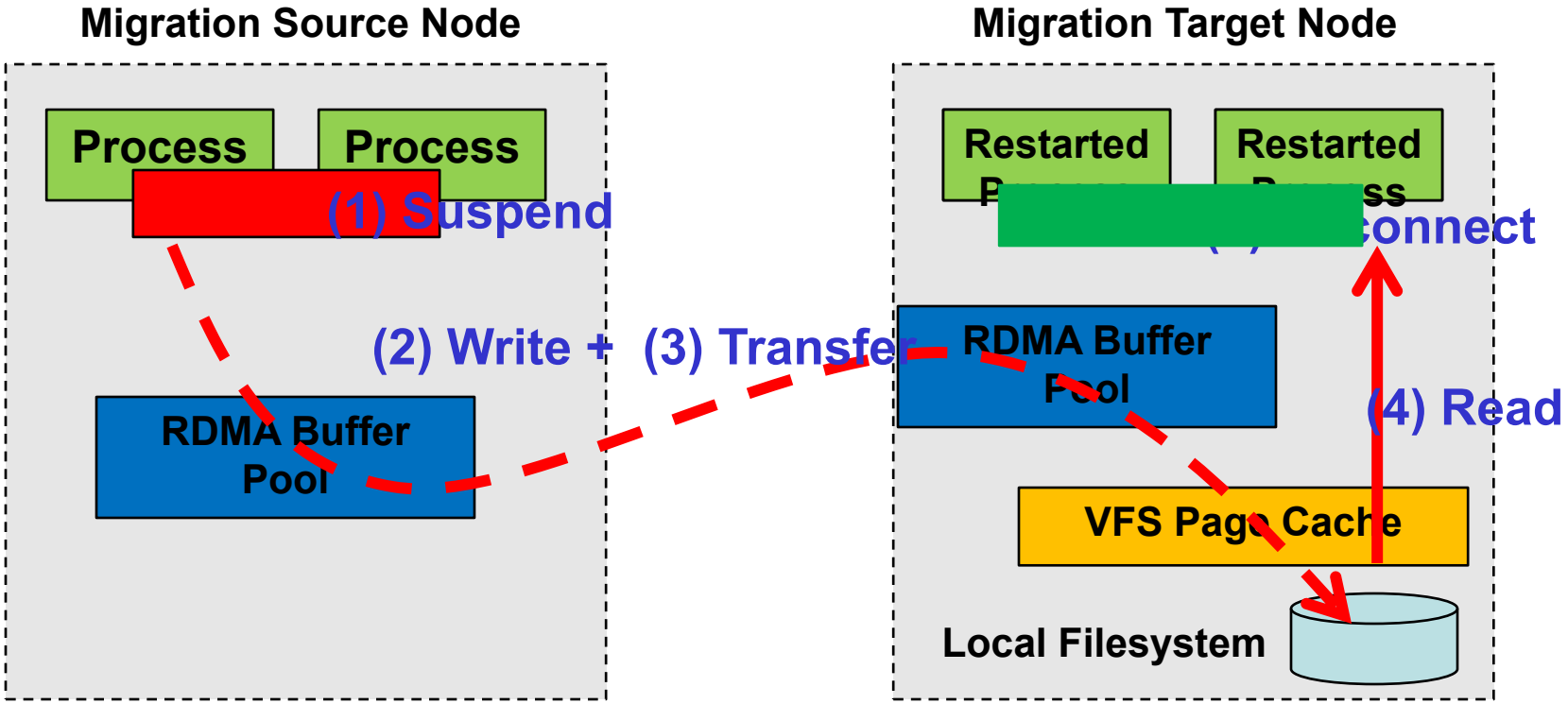
Local Filesystem-based Process Migration (***Local***)

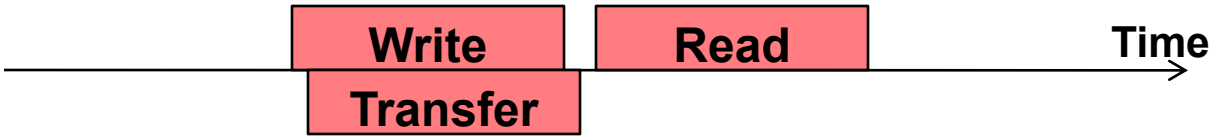


Shared Filesystem-based Process Migration (***Shared***)

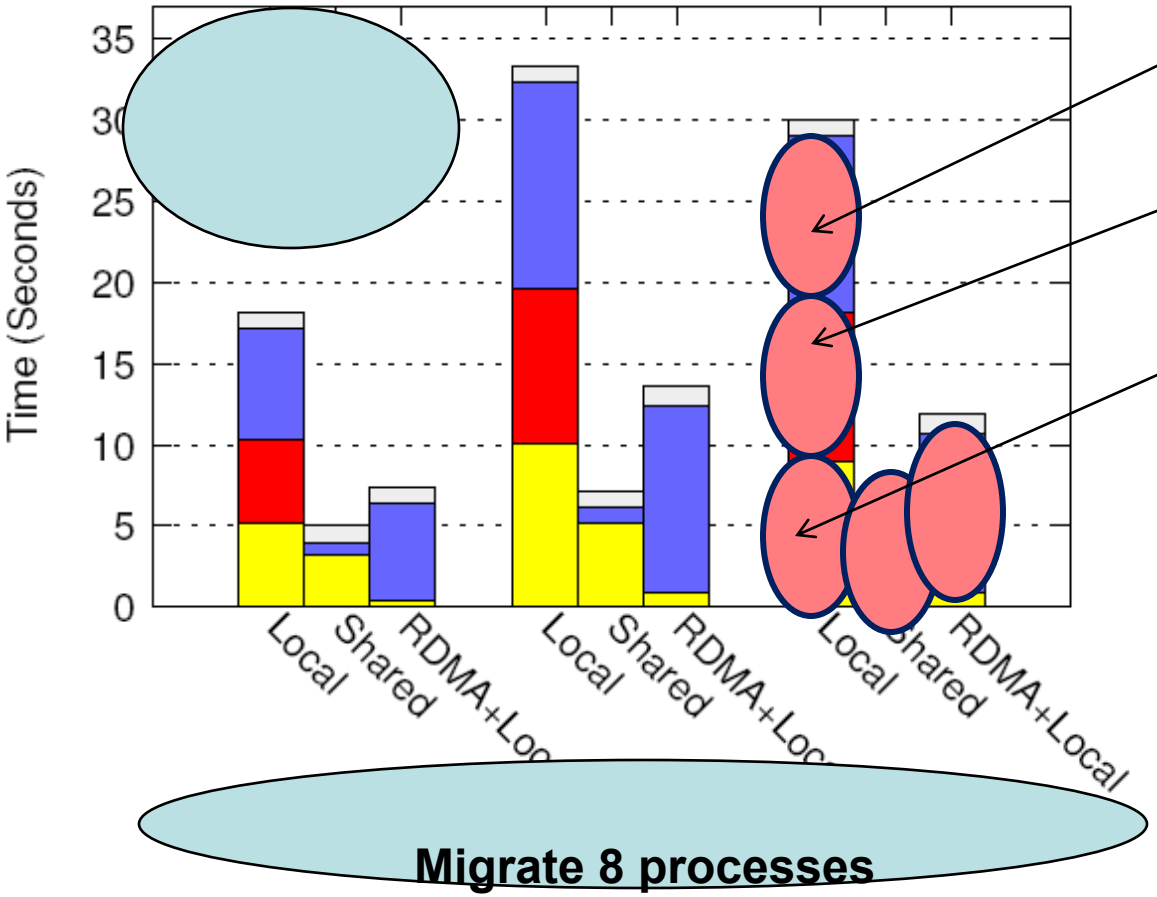


RDMA + Local Filesystem-based Process Migration (***RDMA+Local***)





Profiling Process Migration Time Cost



- **Read image files on target node**
- **Copy checkpoint from source to target**
- **Source node writes checkpoint files**

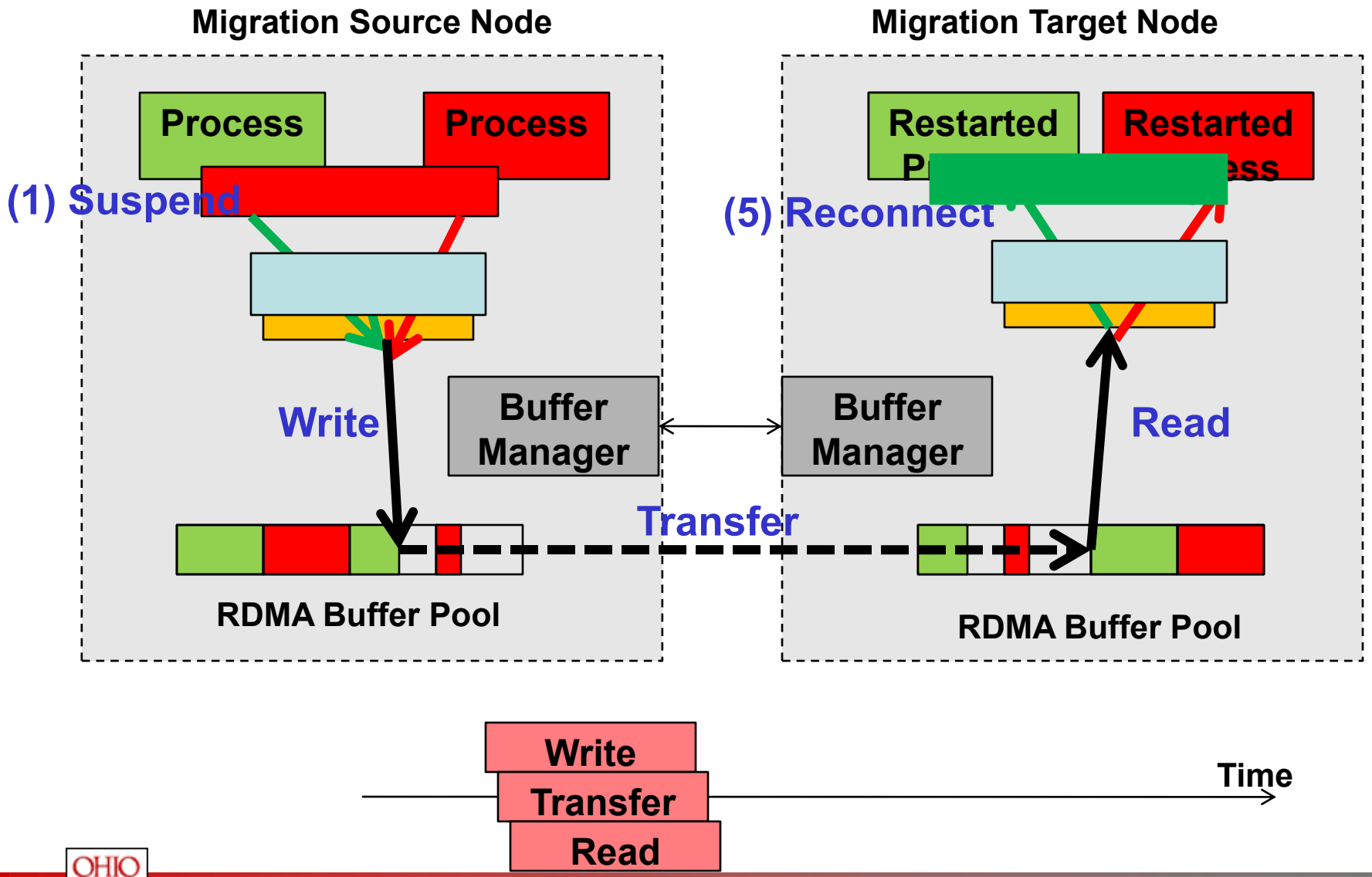
➢ ***All three approaches suffer from IO cost***

➢**Conclusion: All three steps (Write, Transfer, Read) shall be optimized**

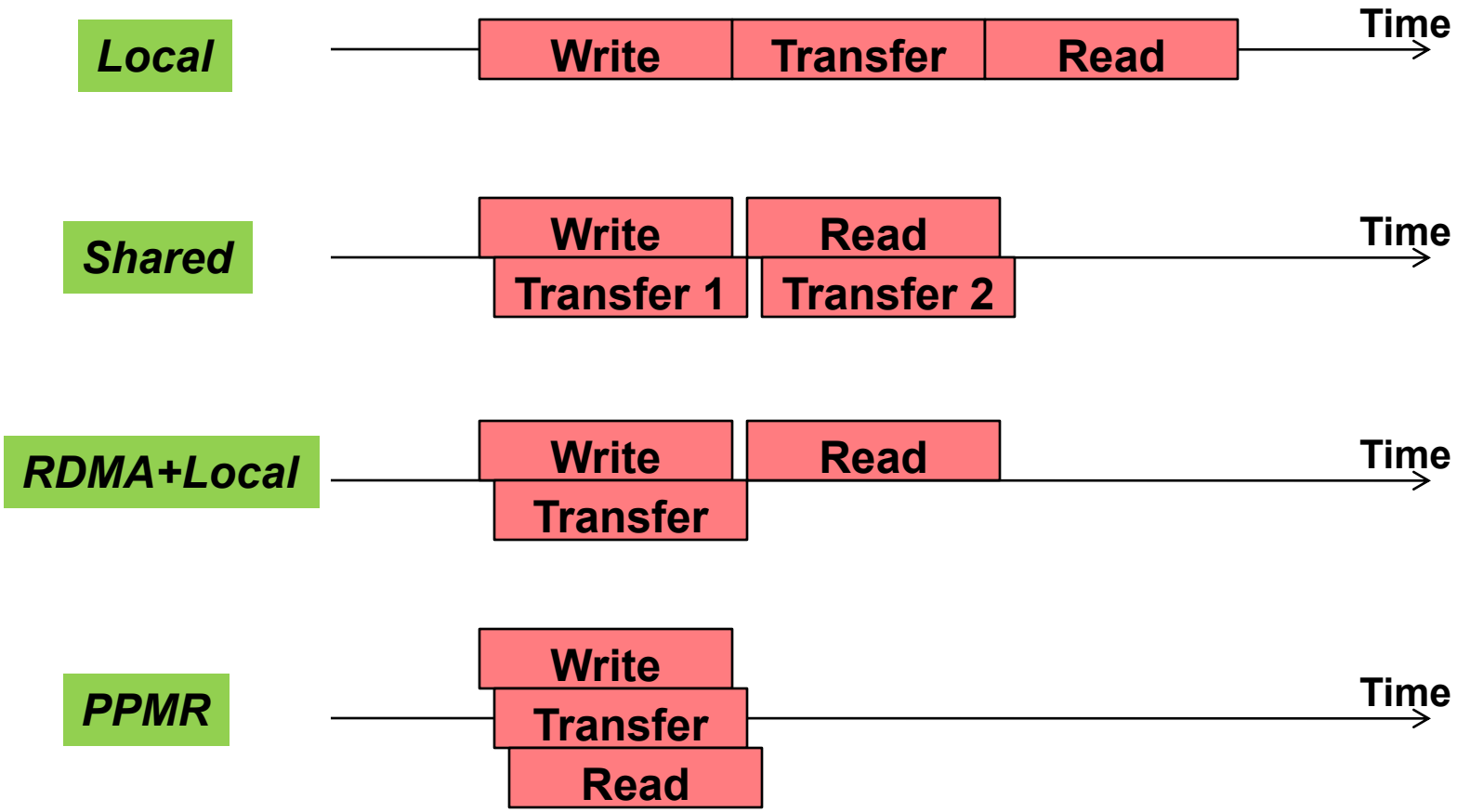
Outline

- Introduction and Motivation
- Profiling Process Migration
- Pipelined Process Migration with RDMA
- Performance Evaluation
- Conclusions and Future Work

Pipelined Process Migration with RDMA (PPMR)



Comparisons



PPMR Design Strategy

- ✓ Fully pipelines the three key steps
  - Write at source node
  - Transfer checkpoint data to target node
  - Read process images
- ✓ Efficient restart mechanism on target node
  - Restart from RDMA data streams

- Design choices
  - Buffer Pool size, Chunk size

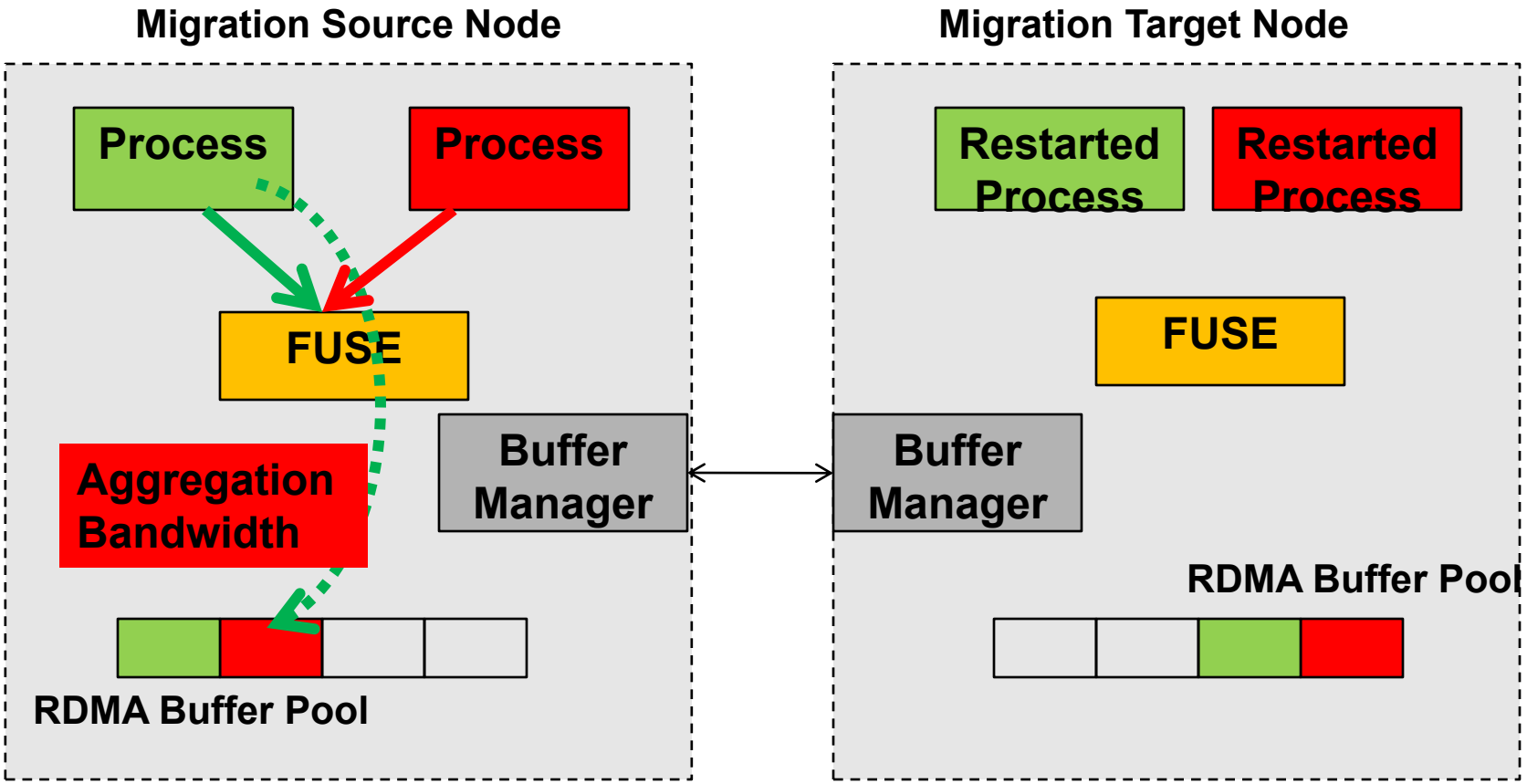
Outline

- Introduction and Motivation
- Profiling Process Migration
- Pipelined Process Migration with RDMA
- Performance Evaluation
- Conclusions and Future Work

Experiment Environment

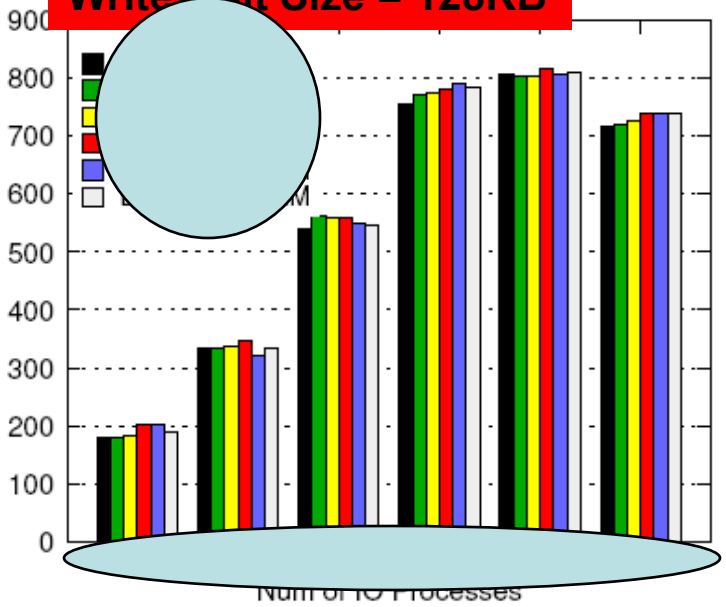
- System setup
  - Linux cluster
    - Dual-socket Quad core Xeon processors, 2.33GHz
    - Nodes are connected by InfiniBand DDR (16Gbps)
    - Linux 2.6.30, FUSE-2.8.5
- NAS parallel Benchmark suite version 3.2.1
    - LU/BT/SP Class C/D input
- MVAPICH2 with Job Migration Framework
    - PPMR
    - Local, Shared, RDMA+Local

Raw Data Bandwidth Test (1)

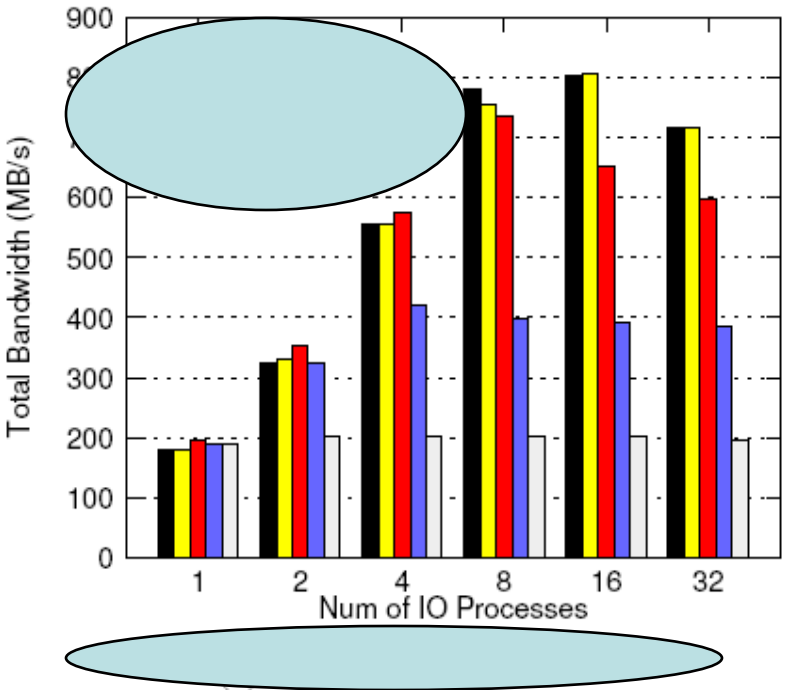


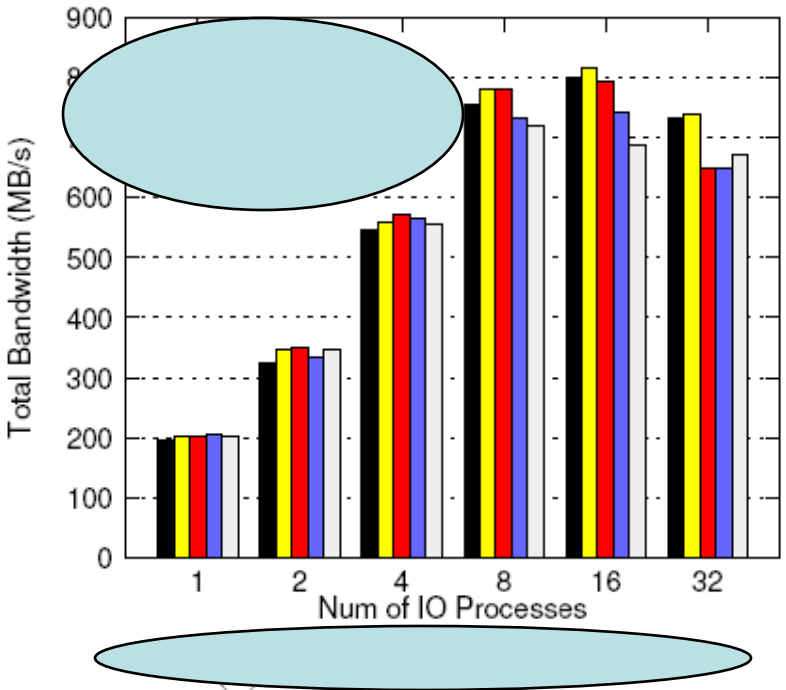
Aggregation Bandwidth

Write Unit Size = 128KB

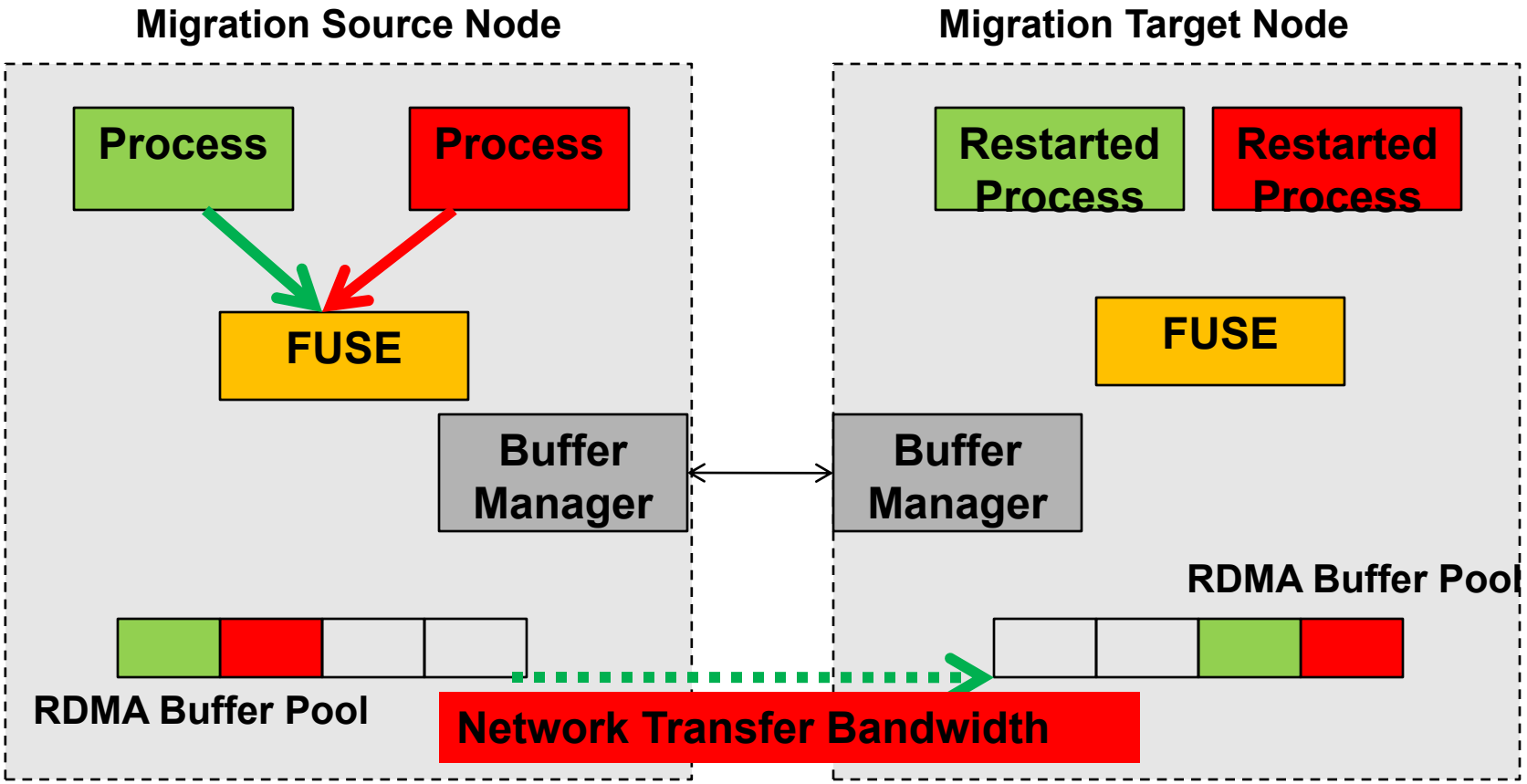


- ✓saturate with 8-16 processes (~800 MB/s)
- ✓Bandwidth determined by FUSE (in-sensitive to buffer pool size)
- ✓Chunk size = 128 KB generally the best

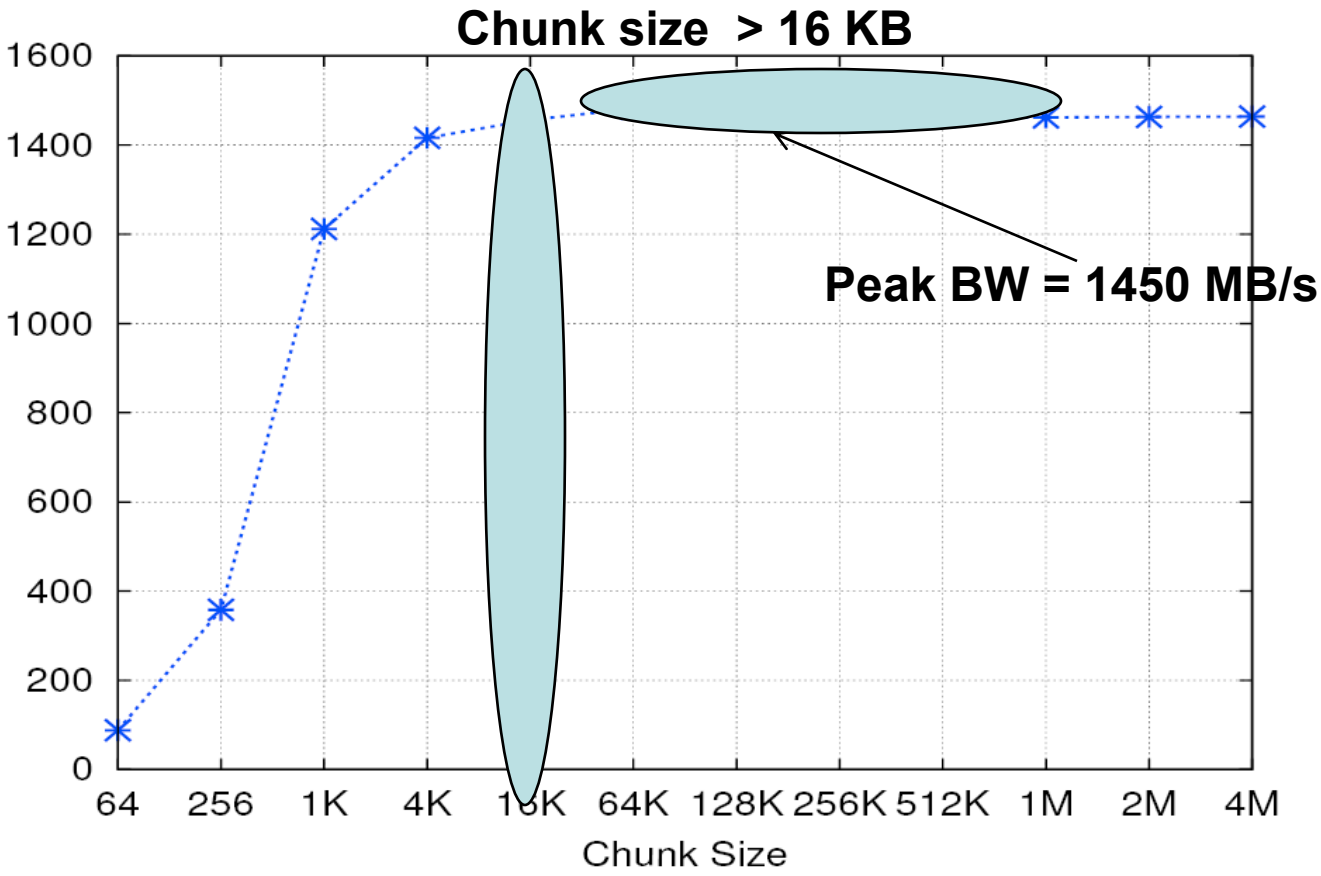




Raw Data Bandwidth Test (2)

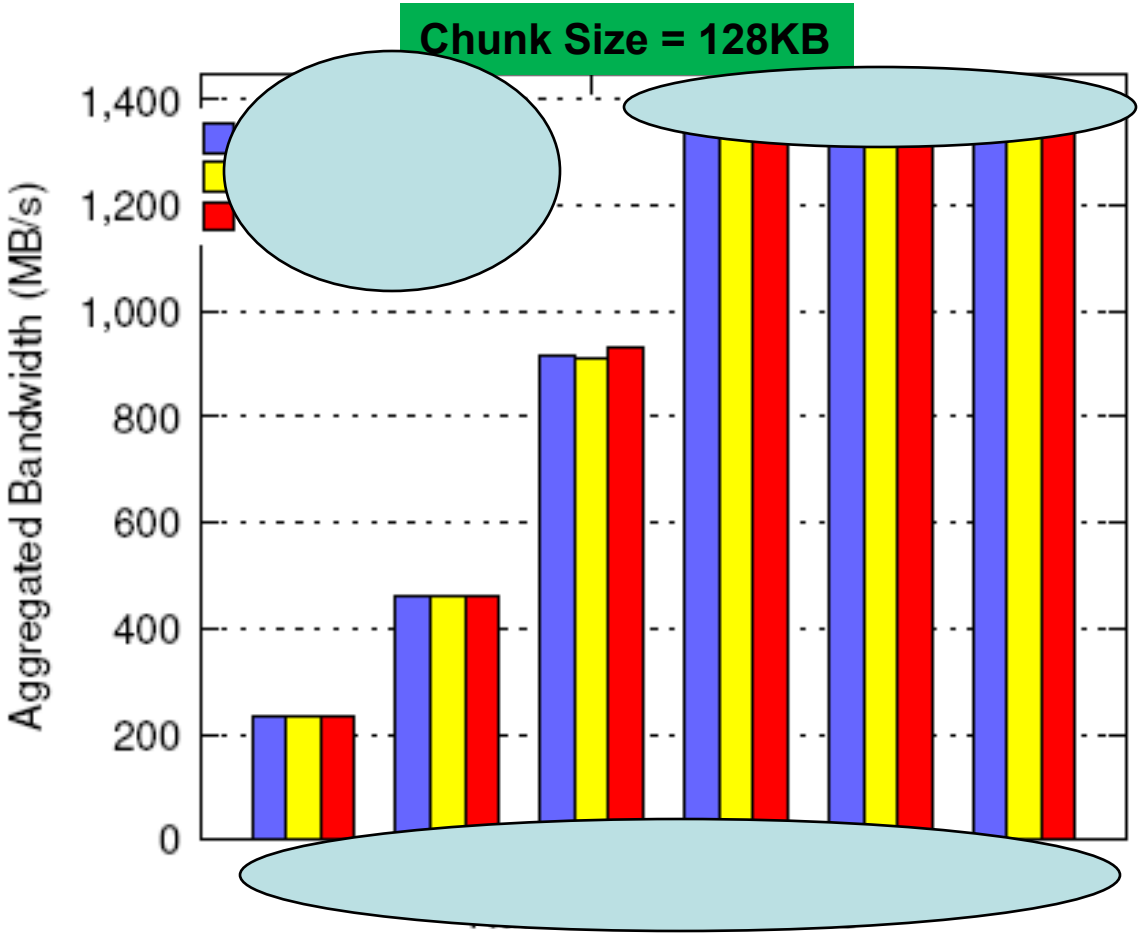


InfiniBand DDR Bandwidth



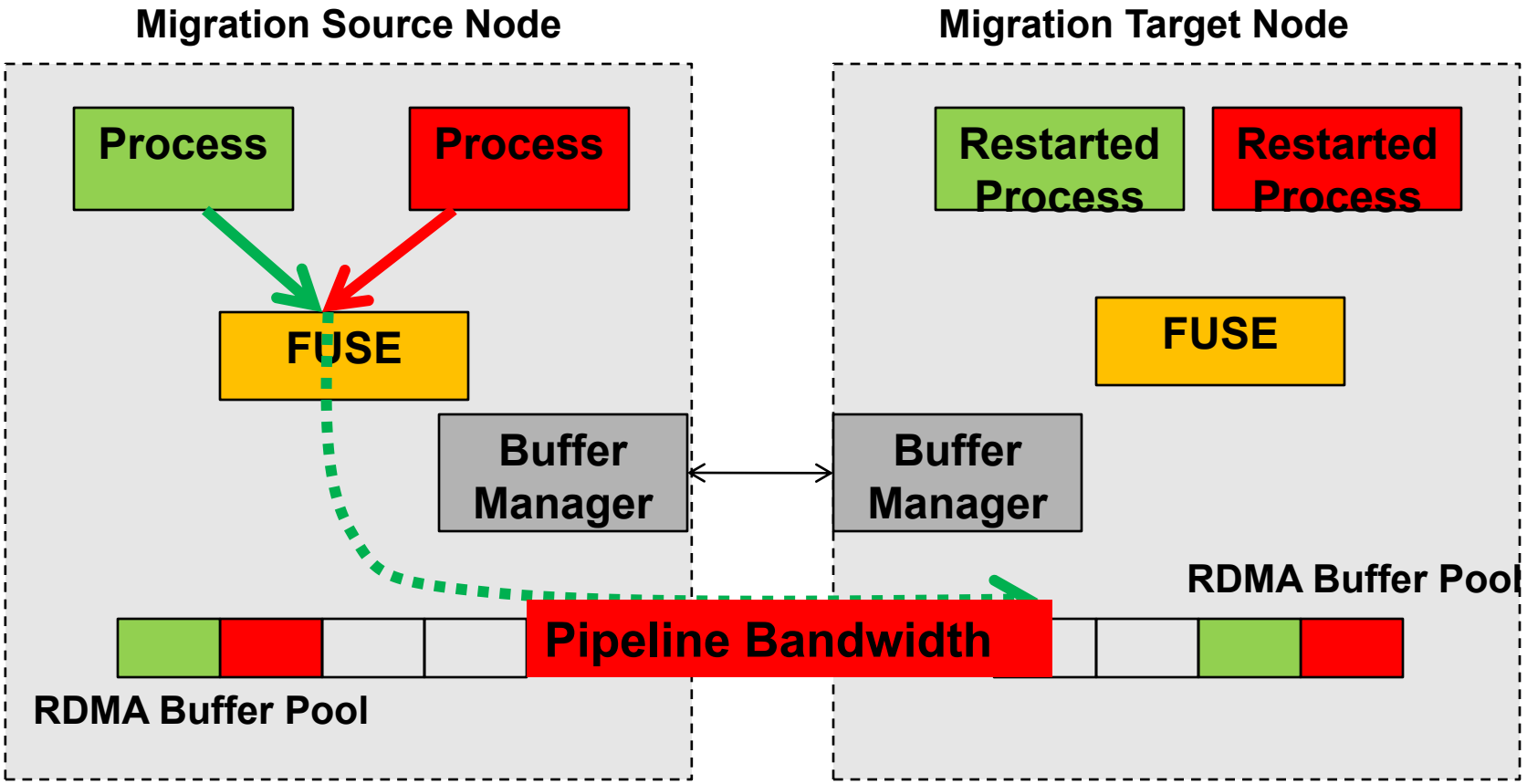
InfiniBand DDR (16Gbps)

Network Transfer Bandwidth

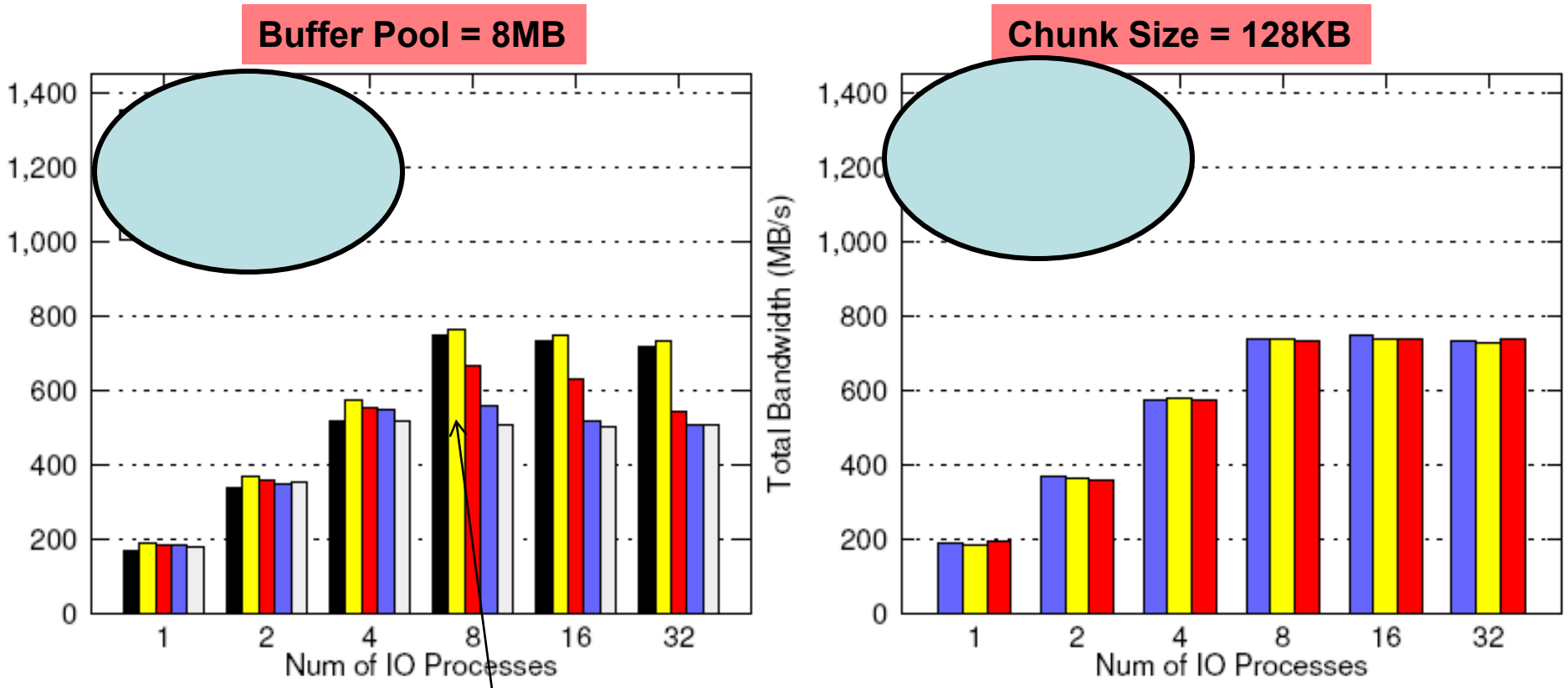


✓Bandwidth in-sensitive to buffer pool size
✓8 IO streams can saturate the network

Raw Data Bandwidth Test (3)

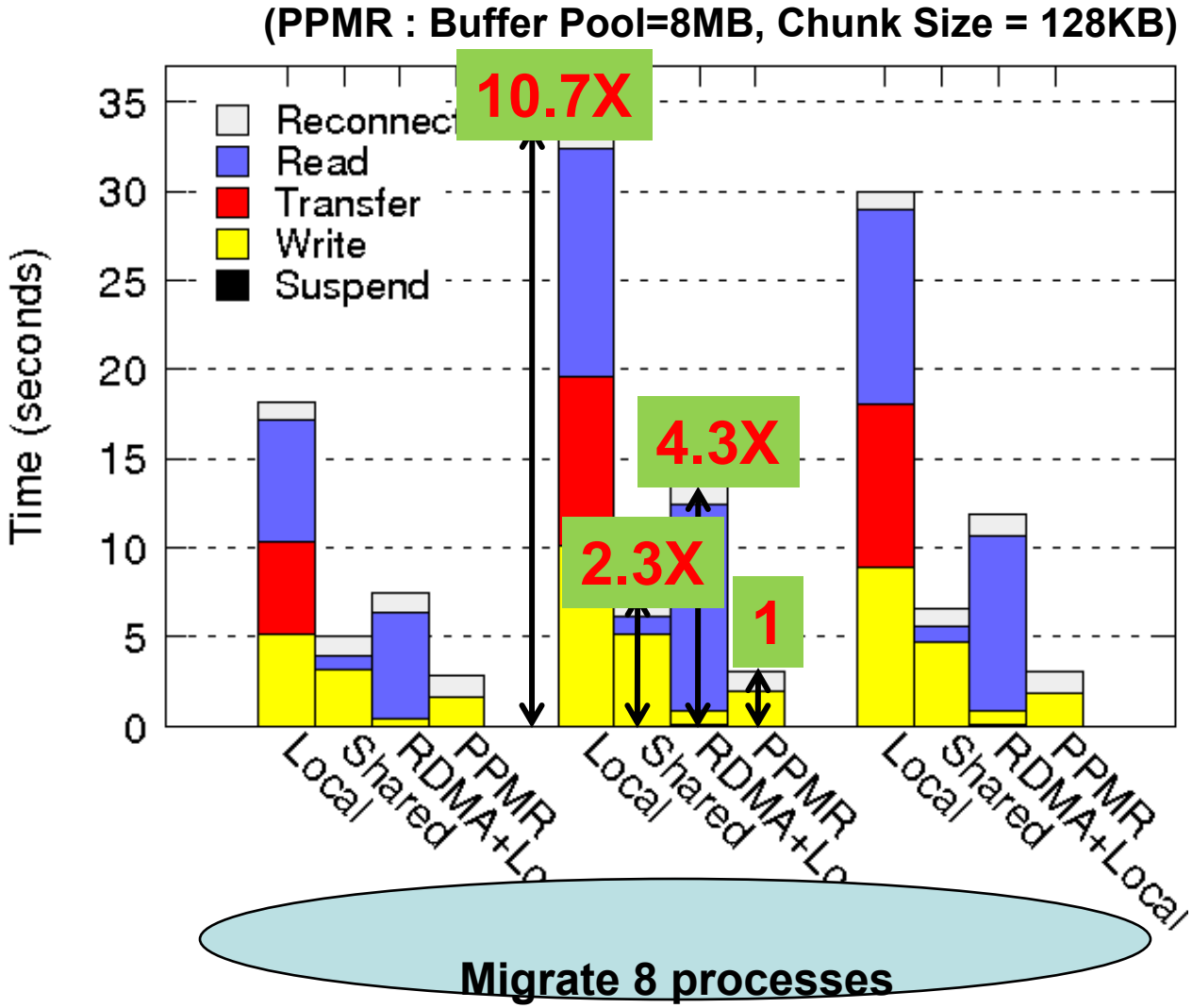


Pipeline Bandwidth

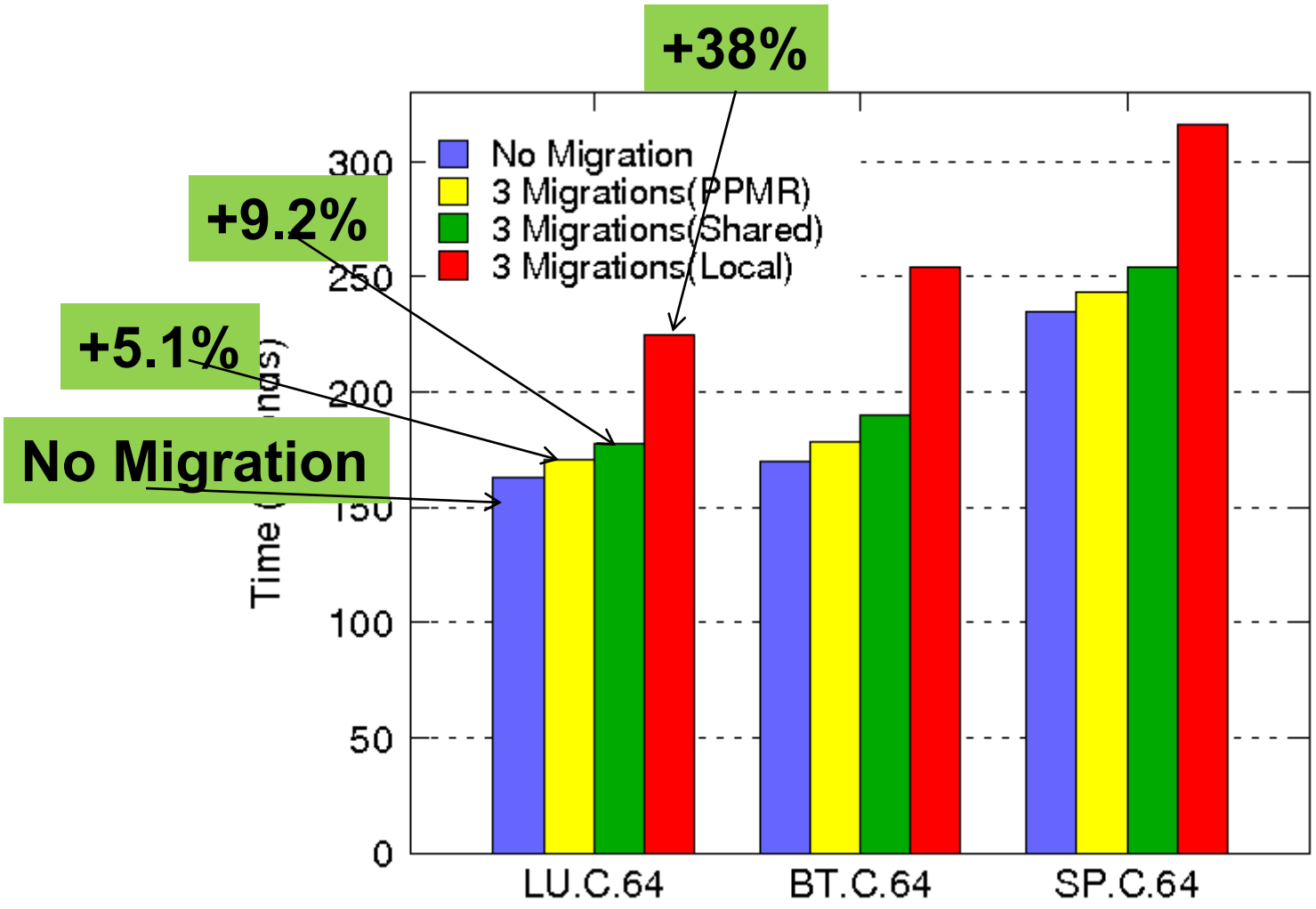


- ✓Determined by Aggregation Bandwidth
- ✓Chunk size = 128 KB generally the best
- ✓Insensitive to buffer pool size

Time to Complete a Process Migration (Lower is Better)



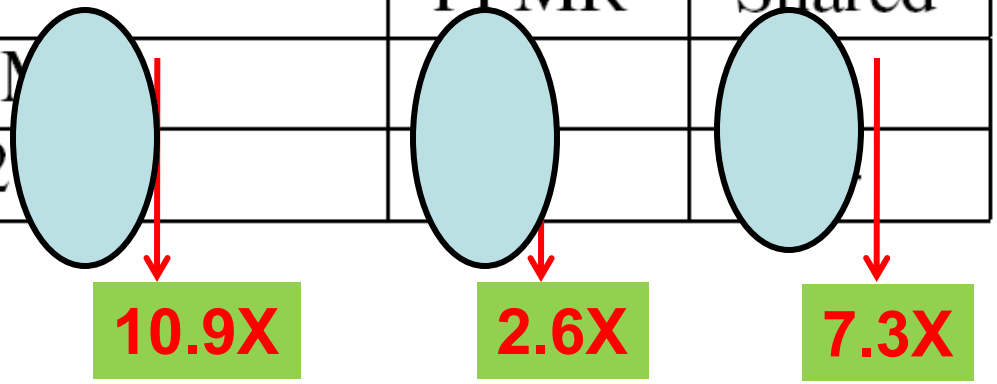
Application Execution Time (Lower is Better)



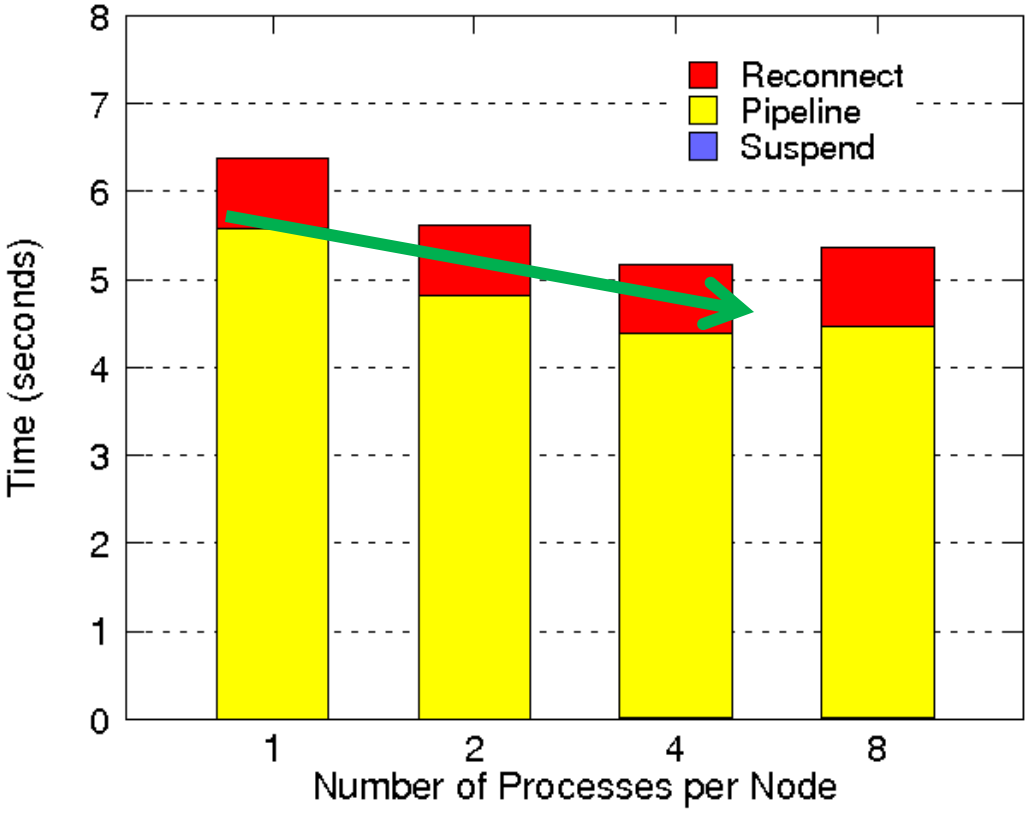
Scalability: Memory Footprint

Migration Time of Different Problem Sizes (64 processes on 8 nodes)

Application	Migrated Data	Time (seconds)	
		PPMR	Shared
BT.C.64	320 M		
BT.D.64	3472		



Scalability: IO Multiplexing



- **Process per Node: 1 ➔ 4**

Better Pipeline bandwidth

LU.D with 8/16/32/64 Processes, 8 Compute nodes.

Migration data = 1500 MB

Outline

- Introduction and Motivation
- Profiling Process Migration
- Pipelined Process Migration with RDMA
- Performance Evaluation
- Conclusions and Future Work

Conclusions

- Process Migration overcomes C/R drawbacks
- Process Migration shall be optimized in its IO path
- Pipelined Process Migration with RDMA (PPMR)
  - Pipelines all steps in the IO path

Software Distribution

- The PPMR design has been released in MVAPICH2 1.7
  - Downloadable from http://mvapich.cse.ohio-state.edu/

Future Work

- How PPMR can benefit general cluster applications
  - Cluster-wide load balancing
  - Server consolidation
- How diskless cluster architecture can utilize PPMR

Thank you!

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

{ouyangx, rajachan, besseron, panda}
@cse.ohio-state.edu

Network-Based Computing Laboratory