



High Performance and Reliable Multicast over Myrinet/GM-2



Weikuan Yu, Darius Buntinas
and
Dhabaleswar K. Panda

Dept of Computer and Info. Science
The Ohio State University

E-mail: {yuw,buntinas,panda}@cis.ohio-state.edu

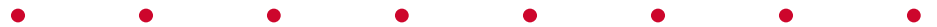




Presentation Outline



- Motivation
- NIC-supported Multicast Schemes
- Our NIC-based Scheme
- Design and Implementation Challenges
- Performance Evaluation
- Conclusions and Future Work





Motivation

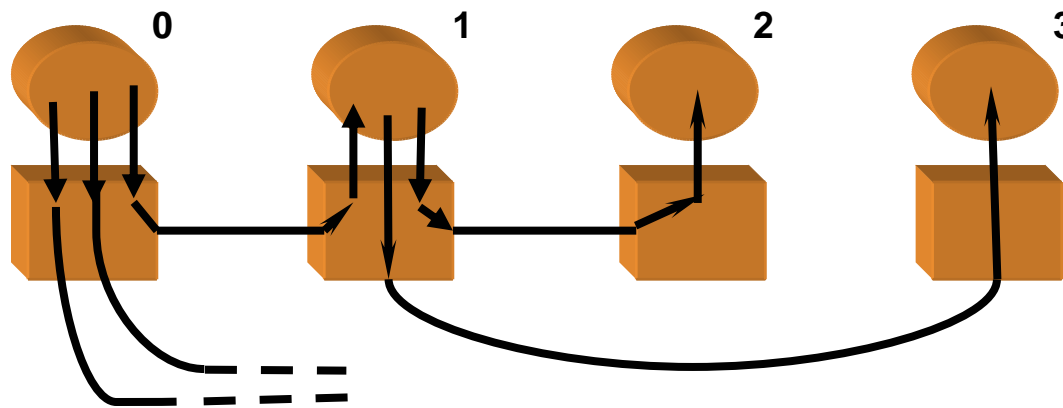


- Some modern NICs have programmable processors
 - Myrinet, Quadrics, Alteon, etc.
- Communication processing can be offloaded from host CPU to NIC
- Efficient communication operations can be supported by NIC, including barrier, broadcast, reduce, etc.

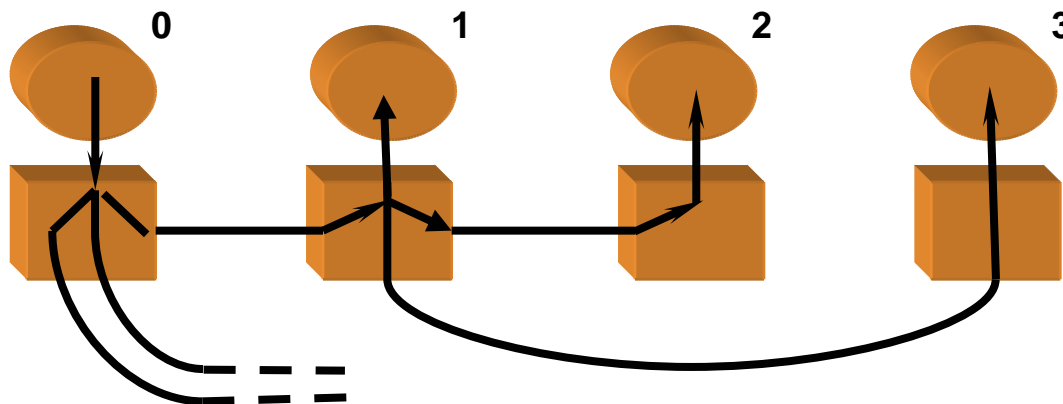


Multicast - Host-based vs. NIC-based

- Host-Based:



- NIC-based:



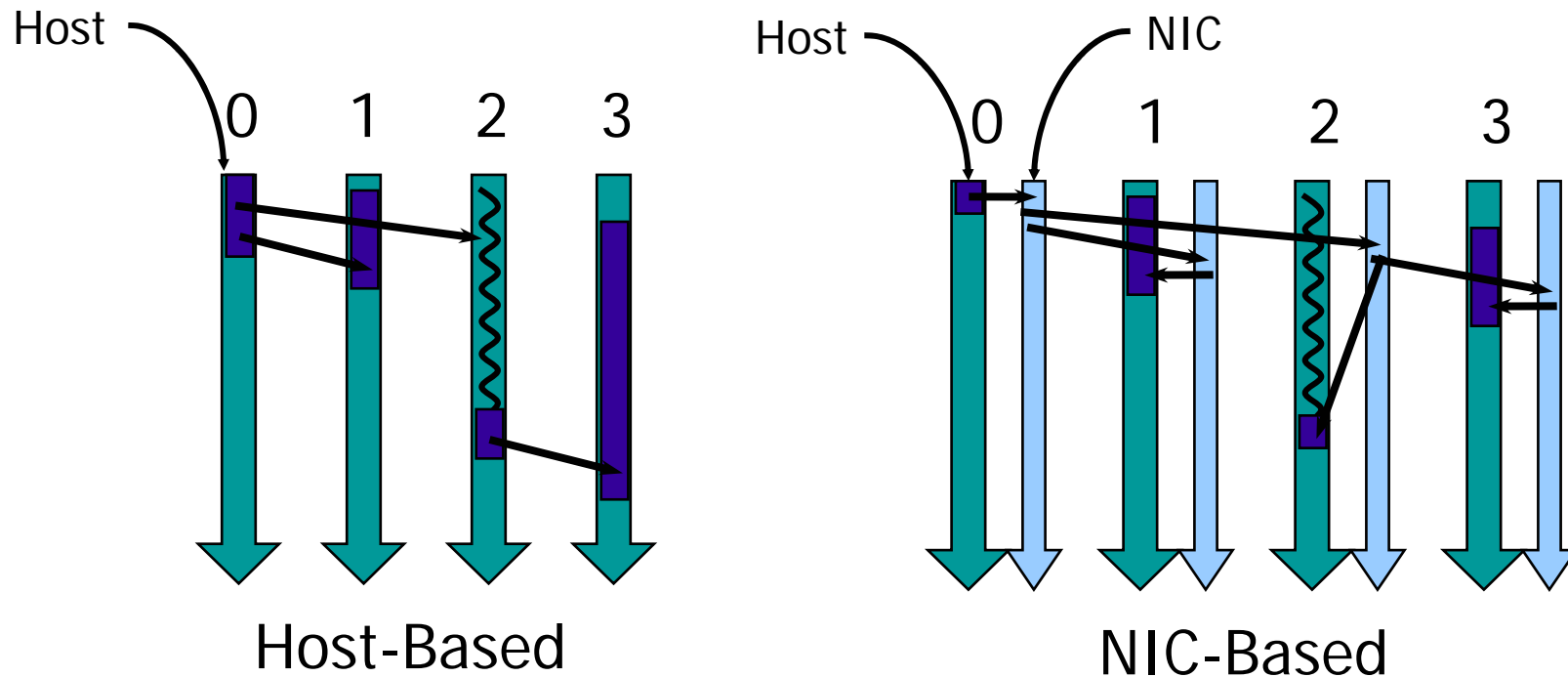


Benefits of NIC-Based Multicast

- Reduced latency
 - Avoiding round-trip on PCI bus
 - Pipelining of multi-packet messages
- Reduced host CPU involvement
 - Forwarding and sending to multiple destinations can be performed at the NIC
- Overlapped computation with the communication
 - Host can compute while NIC performs communication
- Allows for non-blocking or *split-phase* operations
 - Host initiates operation, does other computation, and reads result when needed


Tolerance to Process Skew

- NIC-based multicast allows communication to proceed without involving the user application
- Improved tolerance to process skew
 - Implicit synchronization of broadcast is reduced





Problem Statement

- Myrinet used by many large-scale clusters with GM
 - GM-2 is the next generation protocol
 - A high performance, low latency messaging protocol
 - Good scalability, support thousands of nodes
 - Concurrent protected access to the NIC by several user processes
 - Reliable point-to-point message passing
 - A new structure: Myrinet Packet Descriptor
 - Provides a callback handler
 - Allows efficient repeated transmission of a send packet and the forwarding of a received packet
 - Can we take advantage of the new features of GM-2 to support reliable multicast?
 - Design challenges
 - Performance benefits at the GM-2 and MPI layers
- 



Presentation Outline



- Motivation
- **NIC-supported Multicast Schemes**
- Our NIC-based Scheme
- Design and Implementation Challenges
- Performance Evaluation
- Conclusions and Future Work





Features of Multicast Schemes

- Tree Construction
- Tree information
- Forwarding
- Reliability
- Scalability
- Protection

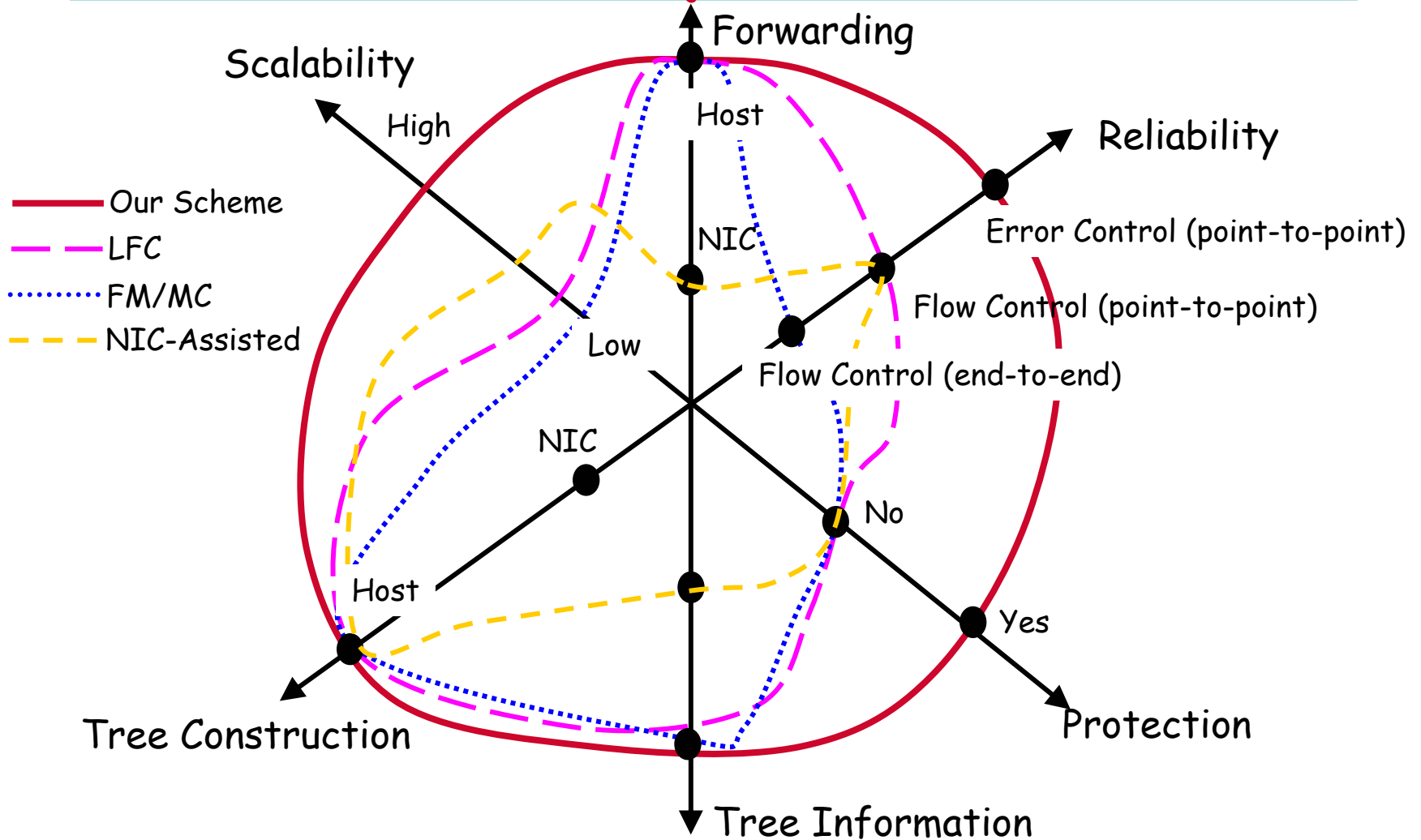




NIC-Supported Multicast Schemes

- **NIC-Assisted Broadcast**
 - Specifies the tree information in the message header
 - Requires intermediate hosts' involvement
- **FM/MC**
 - Provides end-to-end flow control with host-level credits
 - Uses a centralized manager to recycle the credits
- **Link-level Flow Control (LFC)**
 - Uses a link-level point-to-point flow control
 - Deadlock prone

Multicast Schemes - Where Do They Stand?





Presentation Outline



- Motivation
- NIC-supported Multicast Schemes
- **Our NIC-based Scheme**
- Design and Implementation Challenges
- Performance Evaluation
- Conclusions and Future Work





Main Features of the Proposed Scheme



- Efficient tree construction by the host
- Preposting of the spanning tree to the NIC
- NIC-based forwarding
- Provides reliability with error-control
- Achieves Protection and Scalability



Overview of Myrinet/GM

- Myrinet NIC components
 - NIC processor
 - Host DMA engine
 - Network DMA engines (send and recv)
 - CRC/Copy engine (LANai-X)



Send/Receive over Myrinet/GM

- Sending a message
 - Host posts a send request with a send event to the NIC
 - NIC processing
 - Transforms the event into a send token, and queues the token
 - Processes the token and pulls down the packet(s) from the host memory using host DMA engine
 - Transmit engine injects the packet(s) to the network
 - Records the progress of a packet with a send record
 - Removes a send record when the ACK for a packet is received
 - Generates an event to the host when the sending of a message is completed, i.e., no send records left.



Send/Receive over Myrinet/GM

- Receiving a message
 - Posts a receive buffer with a token at the NIC
 - DMAes a received packet to the receive buffer
 - Generates an ACK to the sender of the packet
 - Generates an event to the host when a complete message is received





Critical Resources

- Critical resources involved in sending a message
 - Host events to the NIC
 - NIC processor, host DMA engine, send DMA engine
 - Pipelining of their actions is the key to GM efficiency
 - For the purpose of presentation, we describe both event processing and packets DMAing to the NIC as NIC processing.
- Critical resources involved in receiving a message
 - A receive token at the NIC
 - NIC processor, host DMA engine, receive DMA engine
 - Pipelining of their actions is important at the receiver side

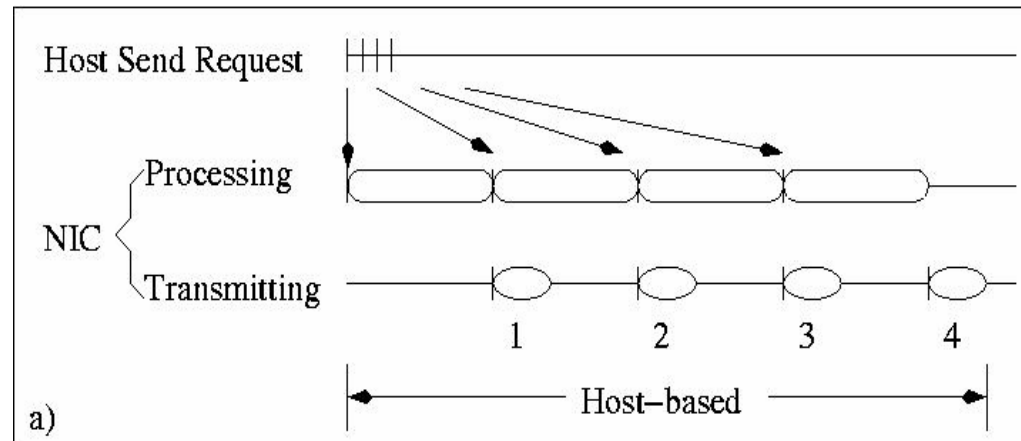
⋮

Two Basic Mechanisms in Our Approach

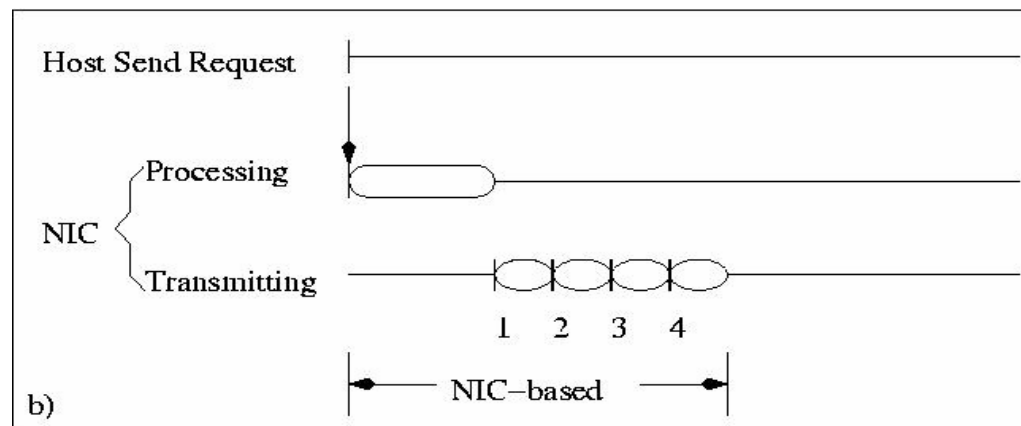
- Multi-send Primitive
- NIC-based Forwarding

Multi-send Primitive

- **Host-Based:**



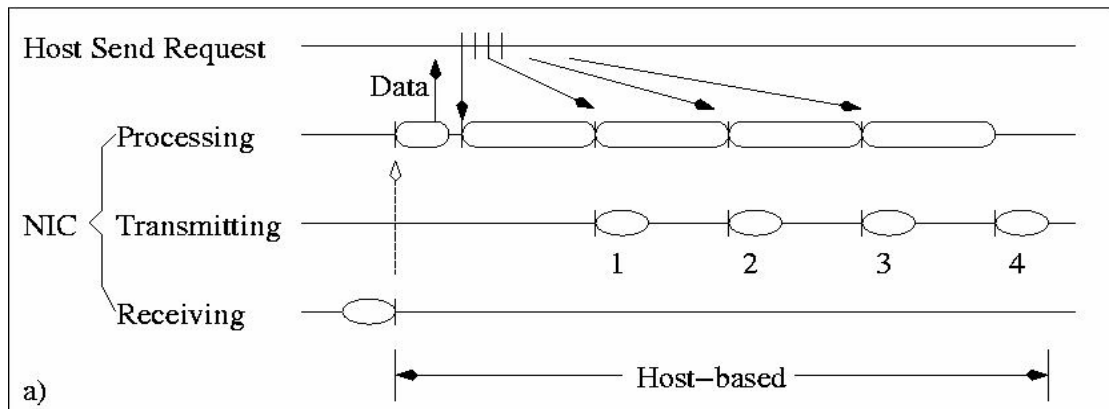
- **NIC-Based:**



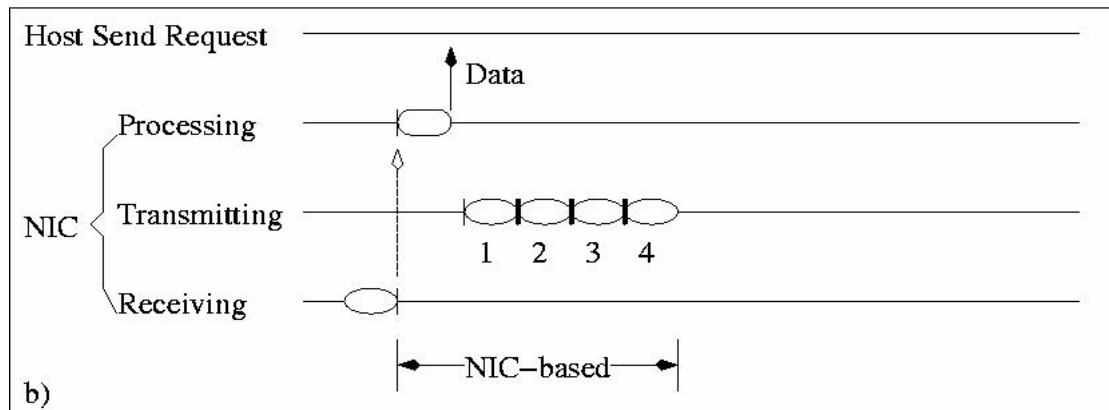
- + Reduced latency
- + Reduced traffic on PCI bus
- Overhead to modify the header and re-queue the packet

Forwarding for Multicast

- **Host-Based:**



- **NIC-Based:**



- + Reduced intermediate host involvement
- + Overlap of computation and broadcast
- + Pipelining of a multi-packet message
- + Tolerance to process skew



Presentation Outline



- Motivation
- NIC-supported Multicast Schemes
- Our NIC-based Scheme
- Design and Implementation Challenges
- Performance Evaluation
- Conclusions and Future Work





Four Major Challenges

- Sending of Multiple Message Replicas
- Forwarding of the received packets
- Reliability and In order delivery
- Deadlock
 - When there is a cyclic dependence on the use of resources among multiple multicast operations






Sending of Multiple Message Replicas

- × Generate a send token for each destination
 - ✓ Save event processing
 - ✓ Easy to implement
 - × Save no PCI traffic
 - × not very beneficial
- ✓ Use the callback handler to resend a packet
 - ✓ Saving token processing
 - ✓ Save host DMA engine from DMAing the same packet, thus PCI bus traffic
 - × Overhead to modify the header and re-queue the packet
 - × Break the ordering imposed by host DMA engine



Forwarding of received packets

- ? Need a send token for the forwarding
 - × Grab a send token from the send token pool
 - ✓ Transform the pre-posted receive token
 - ? Which copy of the message should be kept available for retransmission
 - × The received packet
 - Receive packets are scarce resource
 - Need to be recycled as soon as possible
 - ✓ The copy at the host memory
- 



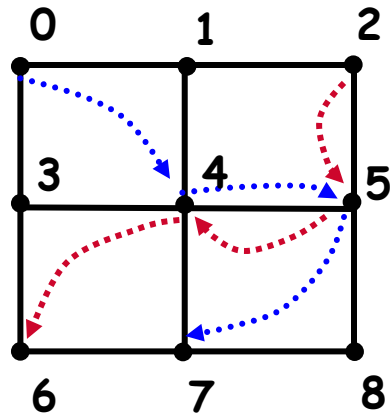
Reliability and In order Delivery



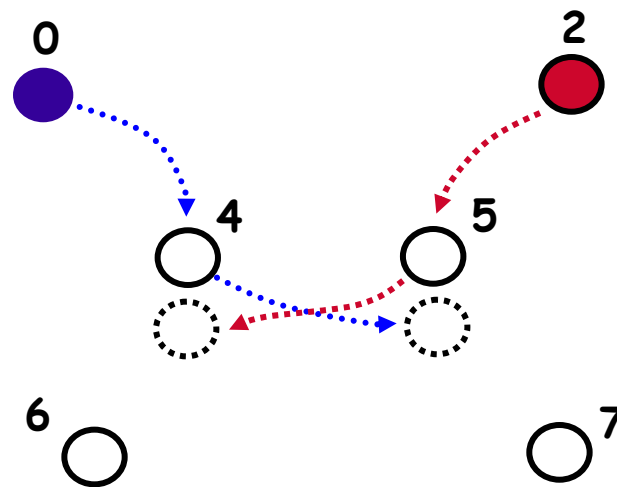
- Use Timeout/retransmission for error-control
- Need to keep track of the ordering of packet delivery for error-control
 - The ordering of the multicast packet delivery w.r.t point-to-point packets is disrupted
 - Use a modified Go-Back-N protocol solely for multicast traffic
 - A receive sequence number for the packets received
 - A send sequence number for the packets sent
 - An array of sequence number for the packets acknowledged by each child

Deadlock

- Cyclic requests of receive tokens are possible among multiple multicast operations
 - Request the receive token for the delivery of one message while holding last the receive token for another multicast message




● Send token ○ Receive token






Deadlock

➤ Solution:

- Order the receivers incrementally according to their GM Network IDs
 - ✓ In the multicast spanning tree, a parent must have a GM Network ID smaller than any child unless it is the root
 - ✓ Roots can not be in a cycle as they holds only send tokens for multicasting
 - ✓ Cyclic requests among receivers are not possible with the incremental ordering
- 



Deadlock

- Cyclic requests of receive tokens are possible among multiple multicast operations
 - Request the receive token for the delivery of one message while holding last the receive token for another multicast message
 - Solution:
 - Order the receivers incrementally according to their GM Network IDs
 - ✓ In the multicast spanning tree, a parent must have a GM Network ID smaller than any child unless it is the root
 - ✓ Roots can not be in a cycle as they holds only send tokens for multicasting
 - ✓ Cyclic requests among receivers are not possible with the incremental ordering
- 



Presentation Outline



- Motivation
- NIC-supported Multicast Schemes
- Our NIC-based Scheme
- Design and Implementation Challenges
- Performance Evaluation
- Conclusions and Future Work





Performance Evaluation



Experiment Testbed:

- 16 nodes
 - Quad-SMP 700MHz Pentium III
 - 64bit/66MHz PCI bus
- Myrinet 2000 network
 - Myrinet PCI64B cards
 - 133MHz LANai 9.1 processor
 - 2MB SRAM
 - 16 ports of a 32 port switch



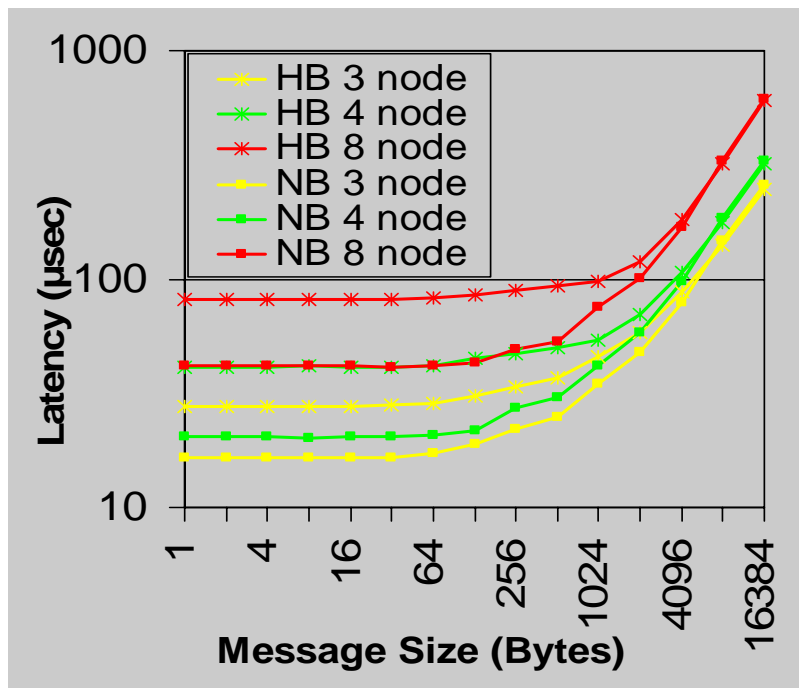


Performance Evaluation

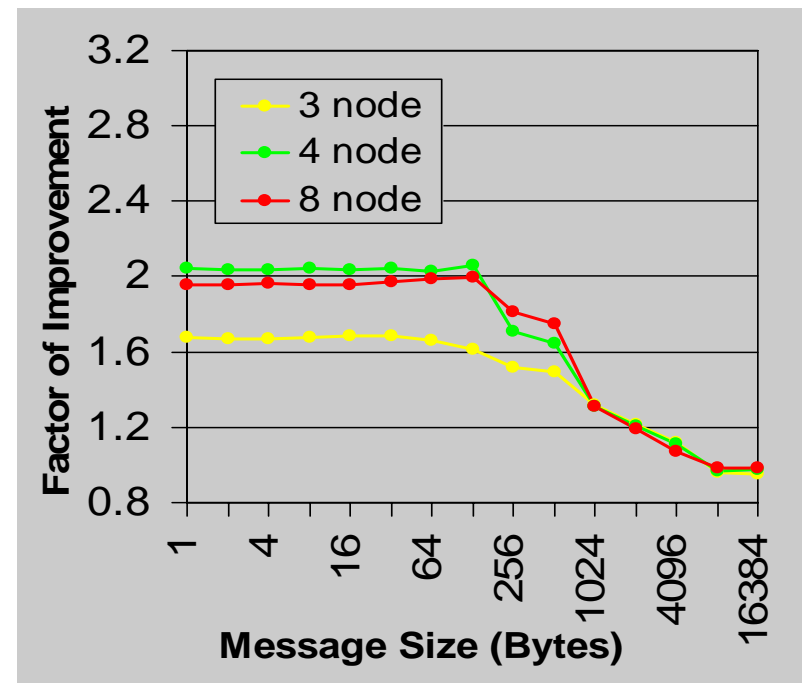
- Multi-send Primitive Performance
- Multicast Performance
- MPI-level Performance
 - Broadcast
 - CPU utilization (the average time to complete a broadcast) in presence of different amounts of average process skew

Multi-send Primitive

Latency



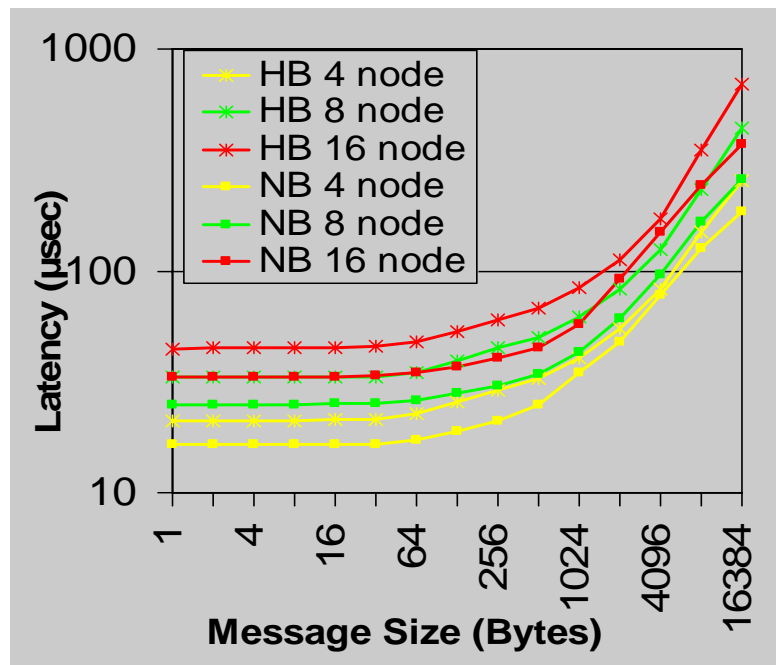
Factor of Improvement



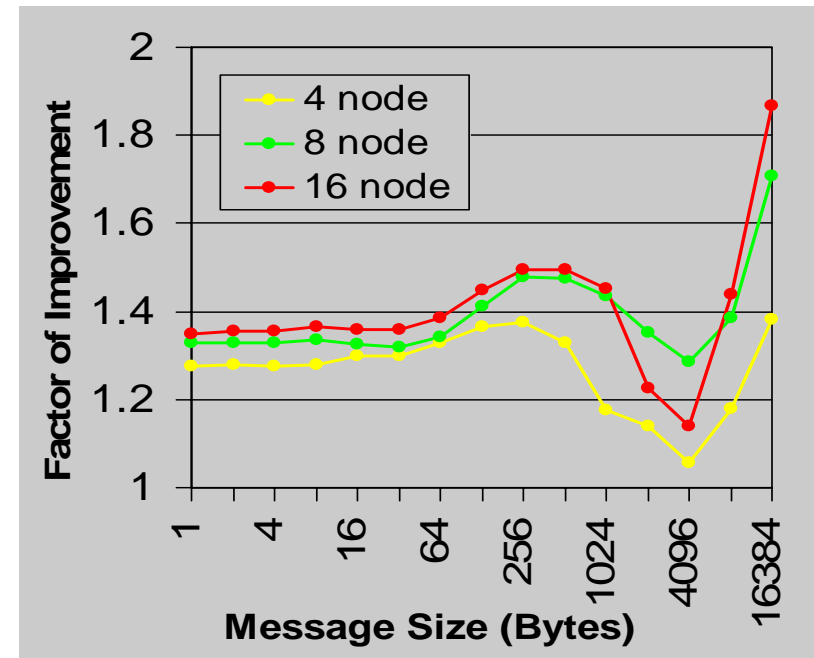
- ✓ An improvement factor up to 2.05 (128 bytes to 4 destinations)
- ✓ Reduced latency for small messages (<1KB) since they benefit from saved processing
- ✗ Large messages (>1KB) can not benefit from NIC-based multisend

NIC-Based Multicast

Latency



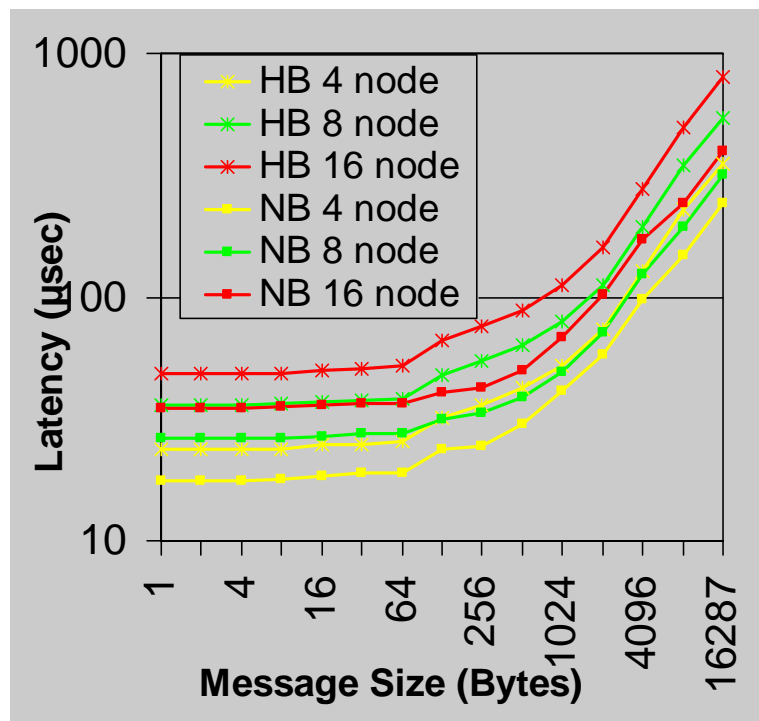
Factor of Improvement



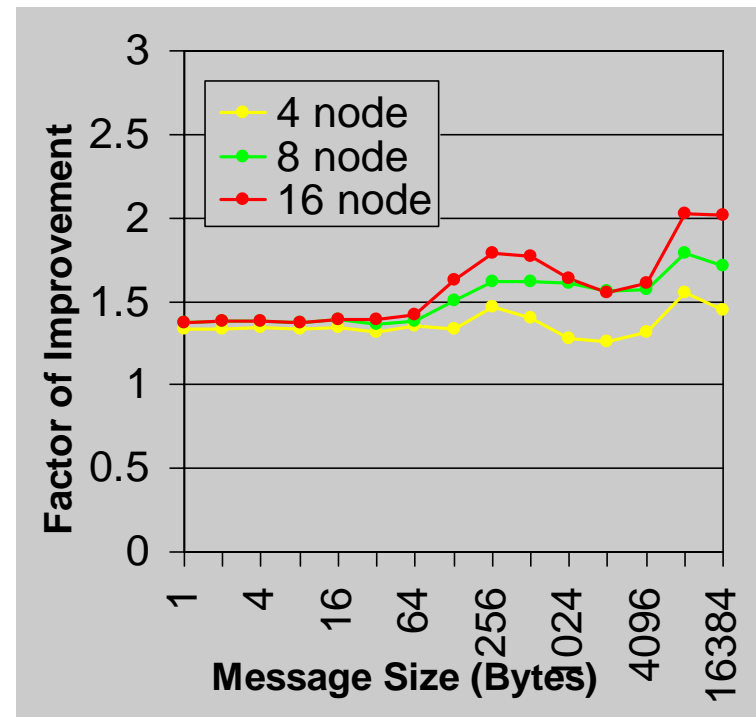
- ✓ Small messages (<1KB) benefit from saved processing
- ✓ Multi-packet messages benefit from pipelining of packets
- ✓ Benefits of reduced host involvement compensate the overhead
- ✓ A factor of improvement of up to 1.86 (16 node)

Performance when Incorporated into MPICH-GM

Broadcast Latency



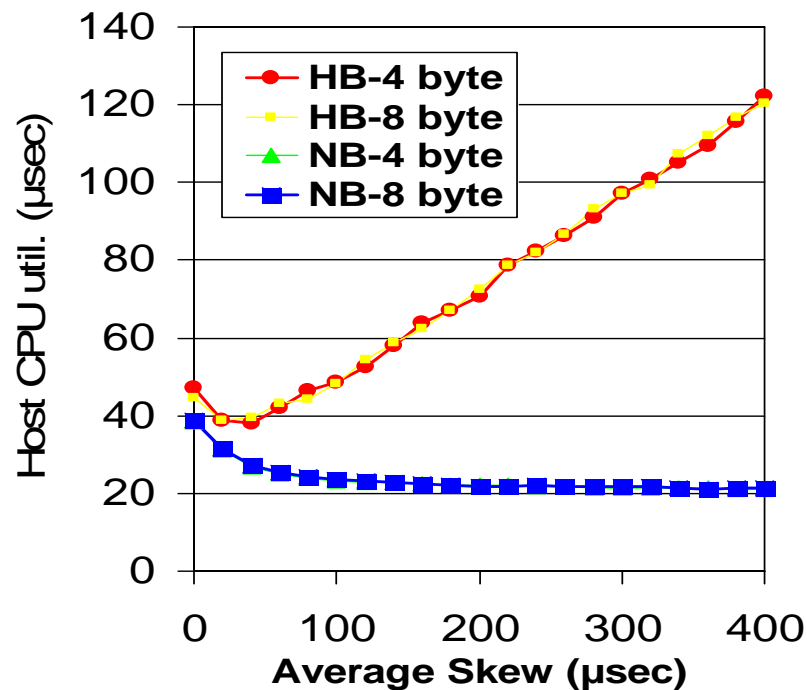
Factor of Improvement



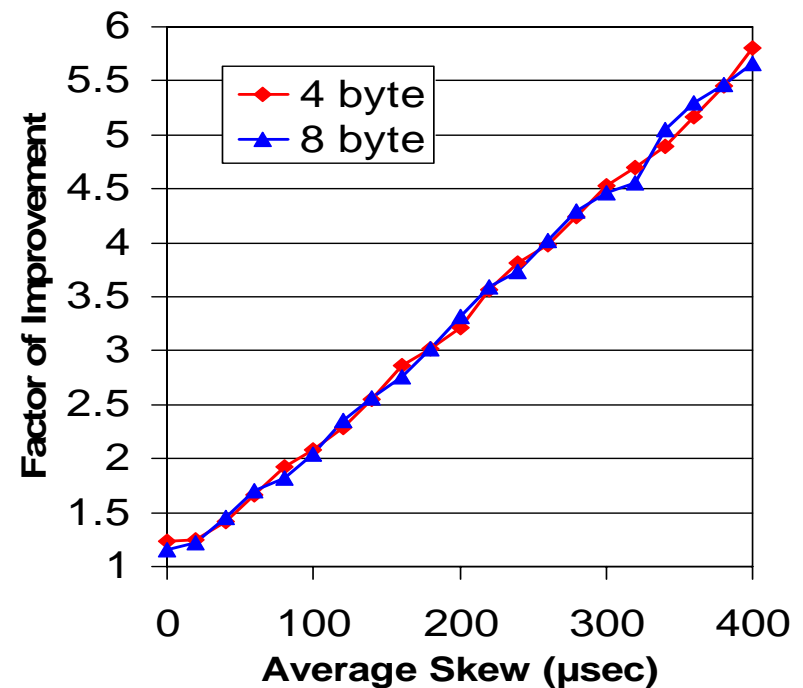
✓ A factor of improvement of up to 2.02 (8KB)

Impact of Process Skew (Broadcast over 16 nodes)

CPU Utilization

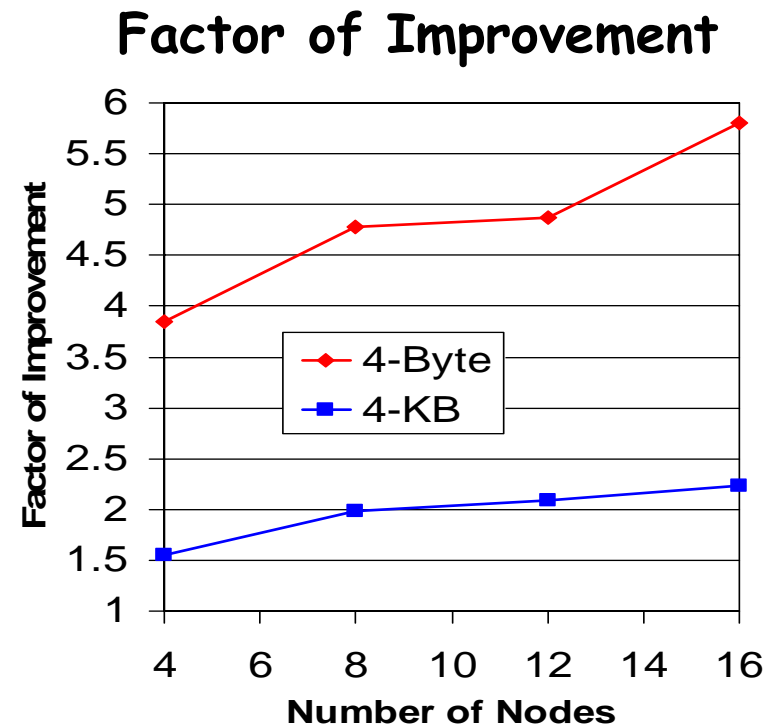
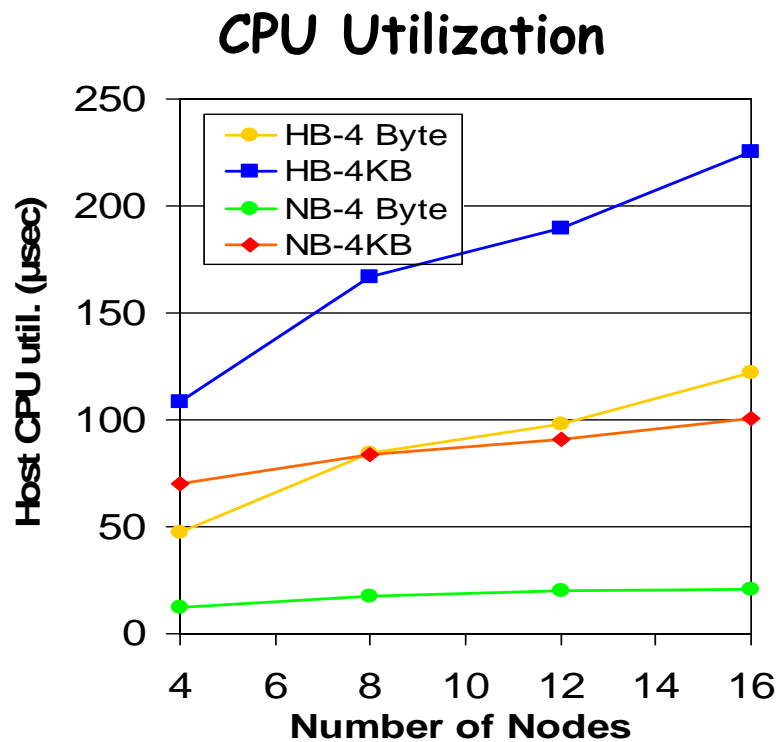


Factor of Improvement



- ✓ 2.1 factor of improvement when skew is only $100\mu s$
- ✓ 5.8 factor of improvement for $400\mu s$ skew

Impact of Process Skew (Broadcast with $400\mu\text{s}$ average skew)



- ✓ Small messages benefit more from the skew tolerance
- ✓ Larger systems benefit more from the skew tolerance



Presentation Outline



- Motivation
- NIC-supported Multicast Schemes
- Our NIC-based Scheme
- Design and Implementation Challenges
- Performance Evaluation
- Conclusions and Future Work





Conclusions

- Characterized important features of NIC-supported multicast schemes
- Proposed a NIC-based Scheme with a suite of salient features
- Designed and implemented an efficient, reliable and NIC-based multicast over Myrinet/GM-2
- Ported the multicast scheme to MPICH-GM
- Explored its benefits to large size systems, such as the tolerance to process skew



Future Work

- Evaluate the scalability of NIC-based multicast on large-scale systems
- Use of NIC-based multicast to support other collective operations, such as Allgather, Allreduce, etc.
- Benefits to collectives with the active NIC support



More Information



NBC

home page

<http://www.cis.ohio-state.edu/~panda/>


<http://nowlab.cis.ohio-state.edu/>

E-mail: {yuw,buntinas,panda}@cis.ohio-state.edu

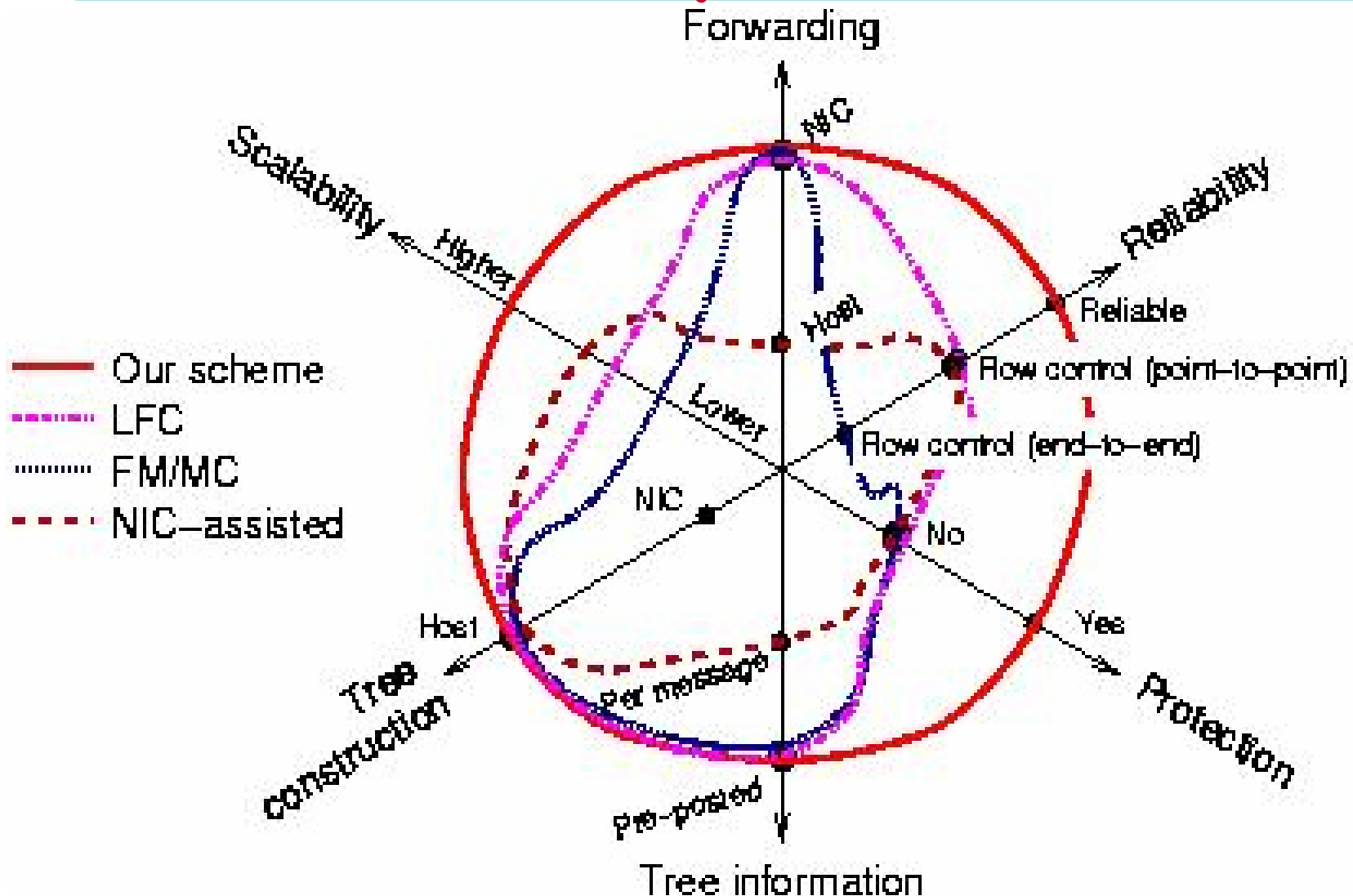




Future Work

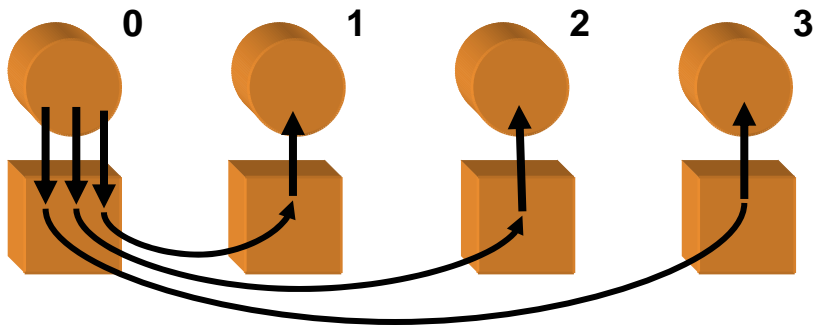
- Evaluate the scalability of NIC-based multicast on large-scale systems
 - Reliable NIC-based multicast with datagram
 - Use of NIC-based multicast to support other collective operations, such as Allgather, Allreduce, etc.
 - Benefits to collectives with the active NIC support
- 

Multicast Schemes - Where Do They Stand?

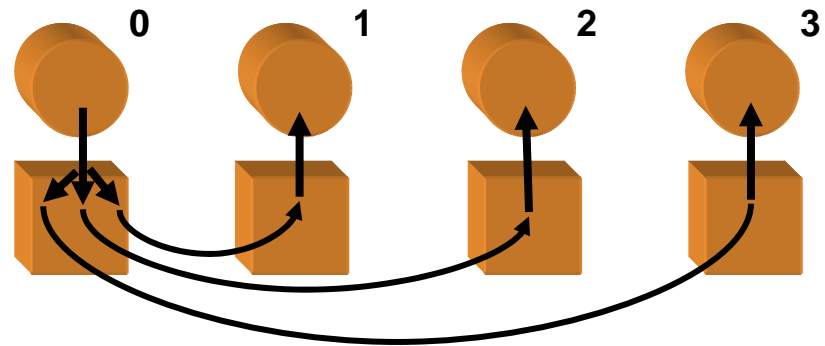


Multi-send Primitive

- Multiple host-based unicasts



- NIC-based multisend



Benefits:

- + Reduced latency
- + Reduced traffic on PCI bus