Scalable and High Performance All-to-All Broadcast over Myrinet/GM

W. Yu, D. Buntinas[+] and D.K. Panda

Argonne National Laboratory[+]
Mathematics and Computer Sci.
buntinas@mcs.anl.gov

Dept of Computer Sci. and Engg.
The Ohio State University
{yuw,panda}@cse.ohio-state.edu

Presentation Outline

- Motivation
  - Benefits of NIC-based collectives
  - NIC-based collective Protocol
  - Feasibility of efficient NIC-based all-to-all broadcast
- Design Challenges and Implementation
- Performance Evaluation
- Conclusions and Future Work

Motivation

- Communication processing can be offloaded from host CPU to NIC programmable processors in modern NICs
  - Myrinet, Quadrics, Alteon, etc.
- Benefits have been exploited in collective operations, such as barrier, broadcast, reduce, etc.
- A NIC-based collective protocol integrated to support all different collectives in a single package

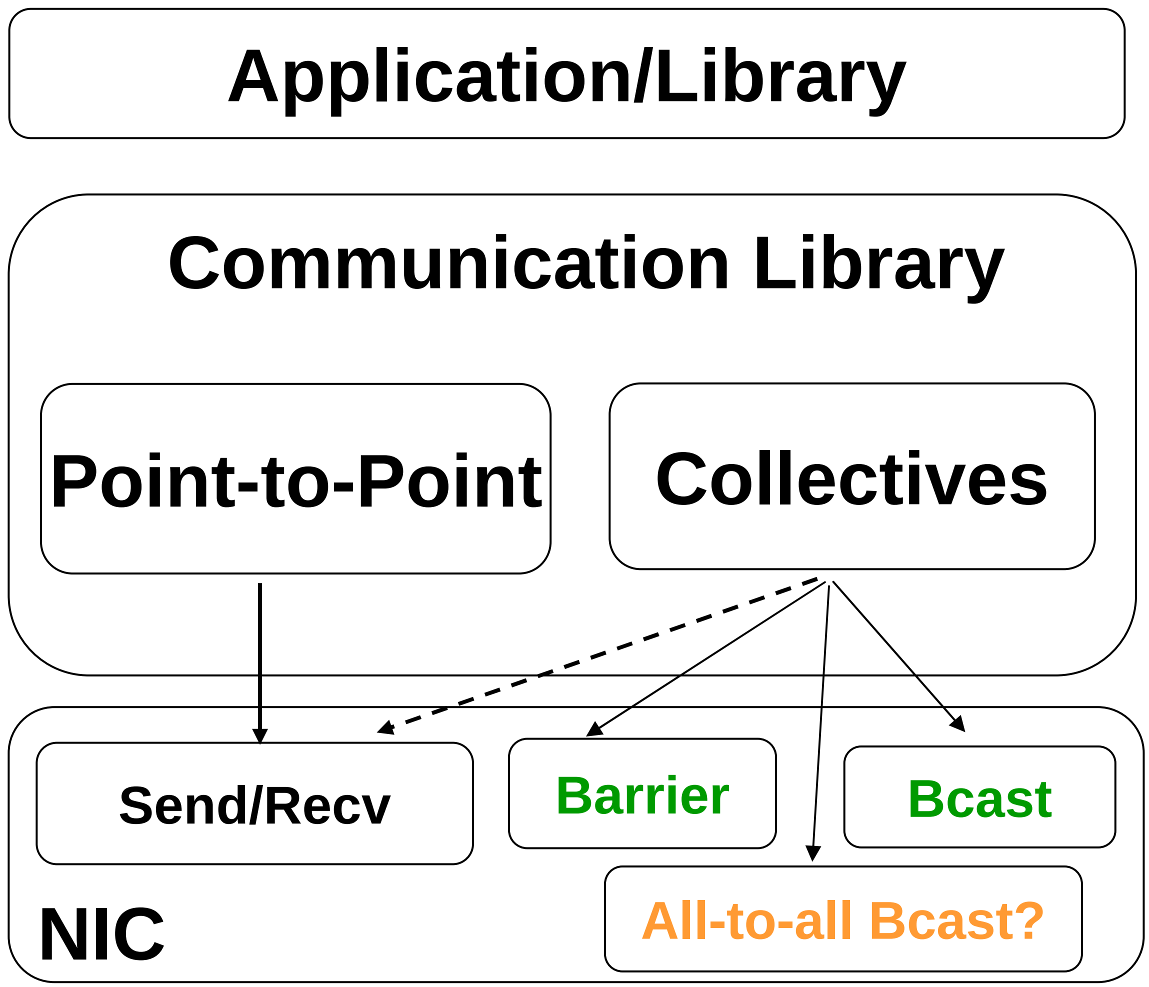
Benefits of NIC-Based Collective Operations

- Reduced latency
  - Avoiding round-trip across PCI bus with forwarding
  - Transparent pipelining of multi-packet messages
- Reduced host CPU involvement
- Overlapped computation with the communication
  - Host can compute while NIC performs communication
  - Allow for *non-blocking* or *split-phase* operations

A NIC-based Collective Protocol

- Offload and integrate a minimal set of Collective operations into the NIC
- Present to the system and library developers an extended collective API, which is built on top of a small set of NIC offloaded collective operations
- Previous work have integrated broadcast and barrier into a collective package

Framework for NIC-based Collective Protocol



All-to-All Broadcast and NIC Offloading

- All-to-All Broadcast
  - Every process broadcasts the same message to others
  - Every process receives messages from all others
  - One of most densely communicating operation
- NIC programmable processors
  - Limited processing power, usually 5-7 times slower
  - Limited memory, up to 64MB, 2 or 8MB over Myrinet
  - Must also handle basic point-to-point traffic

Is an Efficient NIC-based All-to-All Broadcast Feasible?

- Data Communication
  - NIC-based broadcast is beneficial
  - NIC-based data forwarding saves more copying cost
  - NIC-based data aggregation reduces the number of packets
- Synchronization pattern
  - Every process communicates with every other processes, just like barrier, which is beneficial if NIC offloaded
  - Inherent synchronization, easy memory management across different all-to-all broadcast operations
    - two sets of buffers needed, reduced resource constraints

Presentation Outline

- Motivation
- Design Challenges and Implementation
  - Overview of Myrinet
  - Design Challenges
  - Implementation
- Performance Evaluation
- Conclusions and Future Work

Overview of Myrinet/GM

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

Send over Myrinet/GM

- Sending a message
  - Initiation: post a send request as a send event
  - Transmission:
    - Transform event into a send token, and queue it
    - Process the token and prepare the packet(s)
    - Inject the packet(s) to the network
  - Reliability & Retransmission
    - Record the progress of a packet with a send record
    - Remove a send record as ack's coming back
  - Completion
    - when all records are acknowledged.

Receive over Myrinet/GM

- Receiving a message
  - Initiation: post a receive buffer
  - Transmission:
    - DMA packet(s) to the receive buffer
  - Reliability: return an ACK or NACK
  - Completion:
    - Generate a receive event when a message is completed

Design Challenges

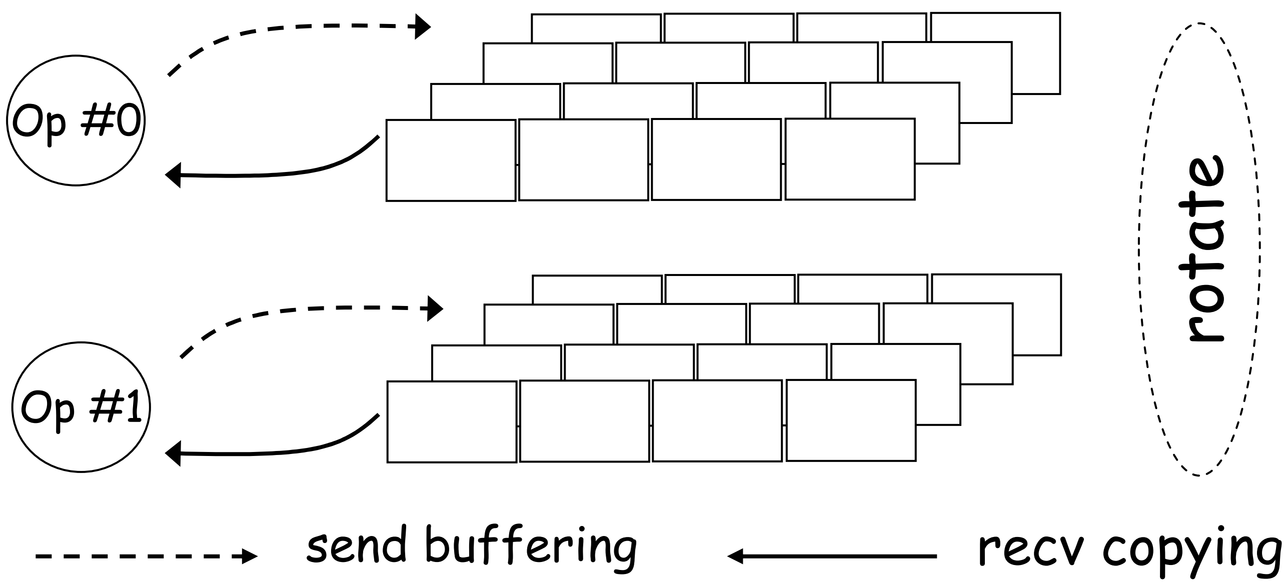
- Topology Management
- Communication processing
- Reliability

Topology Management

- Group Topology needs to be created in advance
  - Fast decision on destination and communication state access
  - Needs to be distributed and scalable
    - Each maintains only a table of the communicating processes
    - Reduce topology information, reduce state maintenance
- Binomial tree is an ideal choice
  - Scalable, 2*logN of entries to maintain for topology
    - $\{ i \square 2^j \bmod N: 0 \square j \square \log N \}$
  - Shared by many collectives including barrier and broadcast
  - Easy bit shifting for topology manipulation

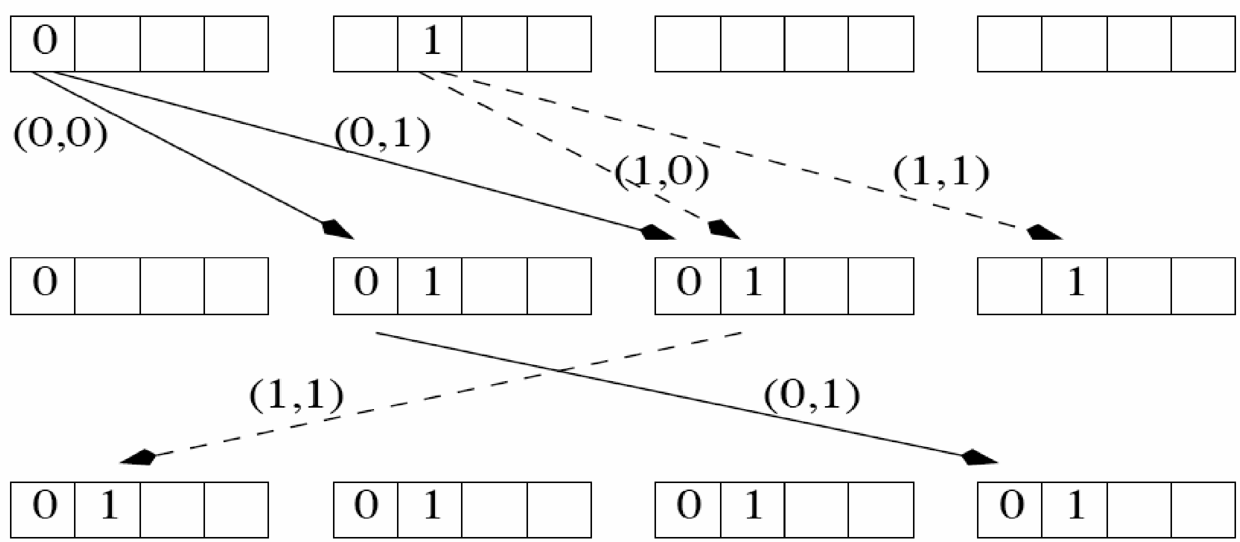
Buffer Management

- System (Host/NIC) buffer is needed
  - Temporary buffering of unexpected packets
  - Data assembly/aggregation at the NIC
  - only two sets of buffers are needed for its synchronous nature



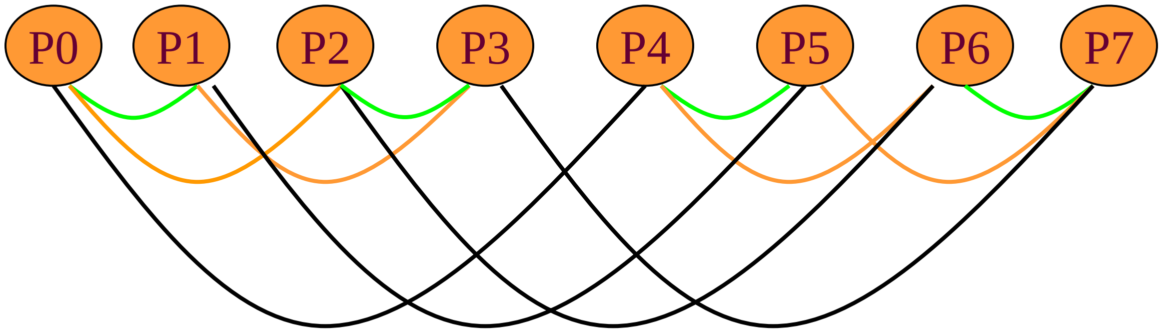
Communication Processing -Concurrent Broadcast

- Each node needs simple and fast method to
  - Broadcasting its own data
  - Forwarding packets that originated from others
- Introducing a flag(i,j) for each packet
  - i being the rank of the originator in the group
  - j the log of the last hop distance
  - Next destination: {|myrank - i| + 2^j}



Communication Processing -Recursive Doubling

- Double the data recursively through pairwise message exchange
- Have benefits of aggregating small packets into larger packets
- No benefits for packets larger than MTU



Reliability

- GM uses sender-driven retransmission based on send records for each packet.
- Propose a receiver-driven retransmission per collective operation, assuming packet corruption is rare
  - NACK to parent/root when not receiving expected packets in time, larger timeout for sentinel purpose
  - Save the acknowledgement packets in normal cases
  - Reduce the resources needed for send-records
- Concurrent Broadcasting
  - Use a bit-vector to keep track of incoming packets
  - Work for message sizes up to one-packet

Implementation

- Based on GM-2.0.3
- Initialized group topology beforehand
- Introduced a new API, gm_gossip()
  - Separate collective queue for collectives
  - Concurrent broadcasting + Recursive doubling
  - Add gm_gossip_recv_event for completion notification

Presentation Outline

- Motivation
- Design and Implementation Challenges
- Performance Evaluation
- Conclusions and Future Work

Performance Evaluation

Experiment Testbed:

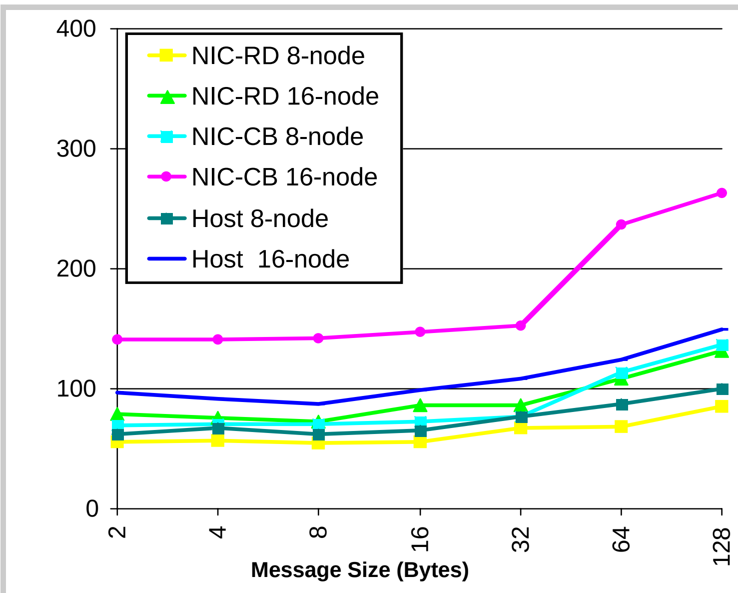
- Myrinet PCI64B cards
  - 133MHz LANai 9.1 processor
  - 2MB SRAM
- 16 ports of a 32 port switch
- Dual-SMP 1GHz Pentium IV

Experimental Results

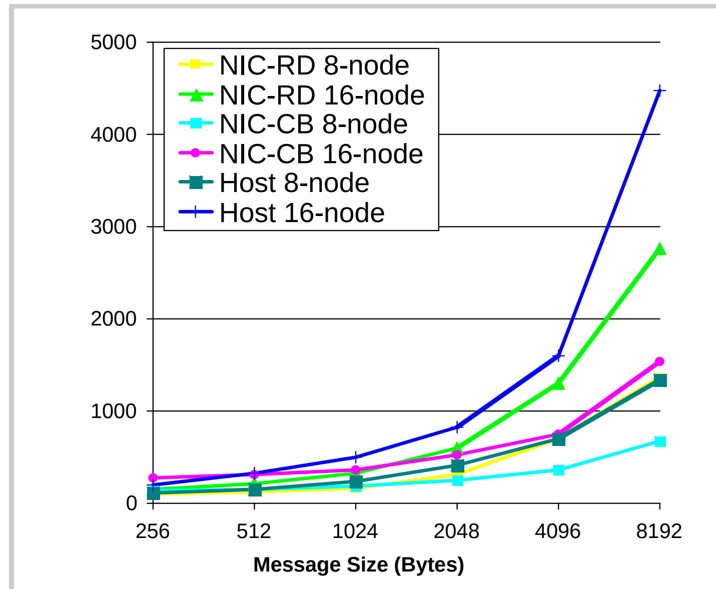
- Latency
- Bandwidth
- Scalability
- Low CPU utilization
- Two NIC-based algorithms are evaluated
  - NIC-RD: Recursive Doubling
  - NIC-DB: Concurrent Broadcasting

Latency

Small Messages



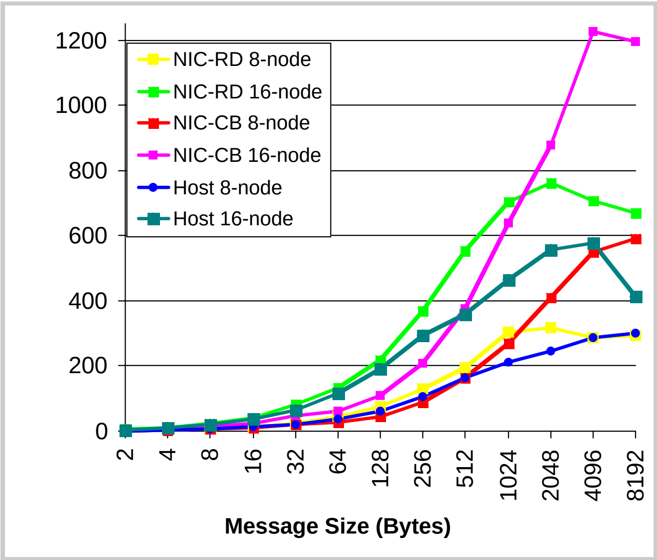
Large Messages



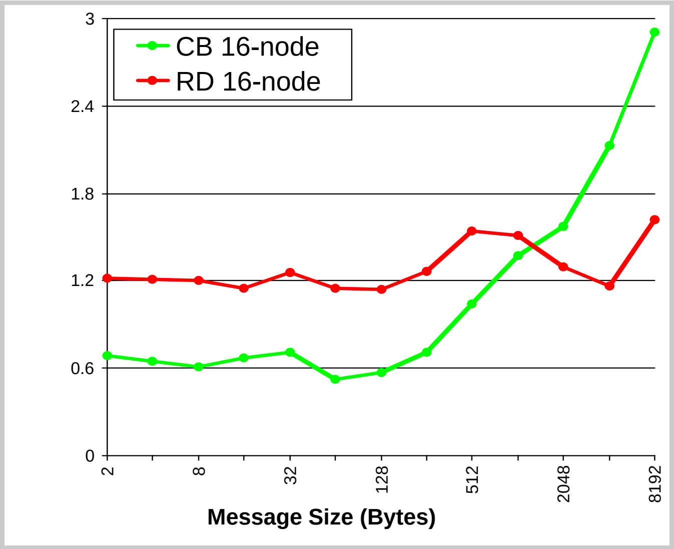
- NIC-RD performs the best for Small messages, being able to aggregate messages into larger packets
- NIC-CB performs the best for large messages, being able to forward packet and reduce copying cost across PCI-bus

Bandwidth

Bandwidth

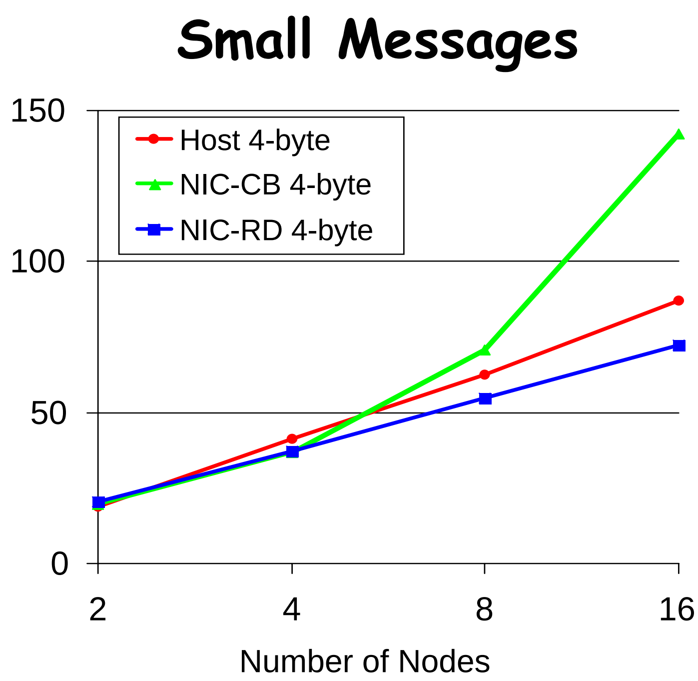


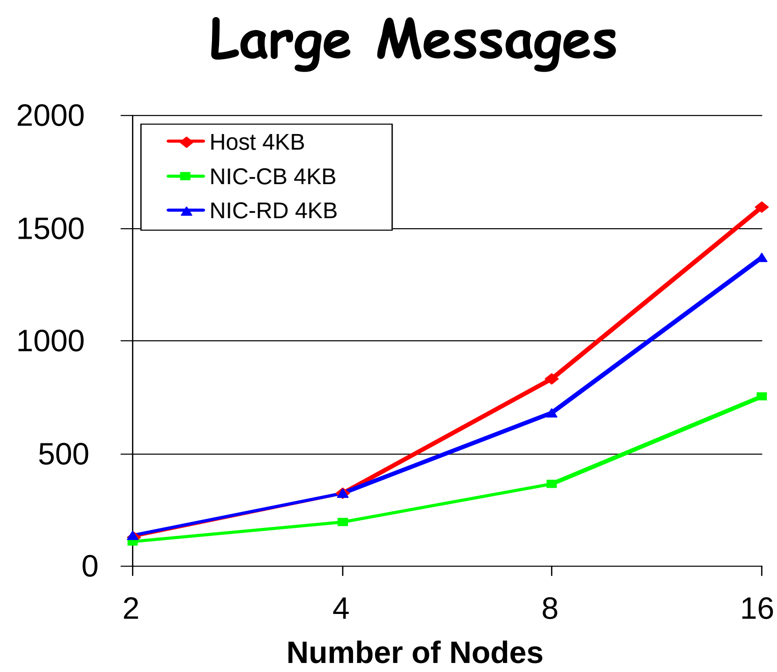
Factor of Improvement



- Both NIC-RD and NIC-CB provides better bandwidth than the host-based all-to-all broadcast for large messages
- NIC-CB suffers for small messages because of processing $O(N^2)$ packets, but performs the best for large messages with benefits of packet forwarding

Scalability

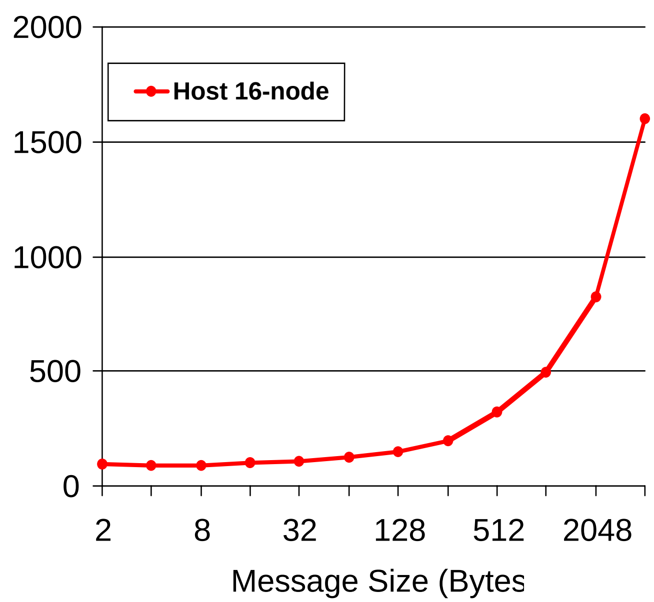




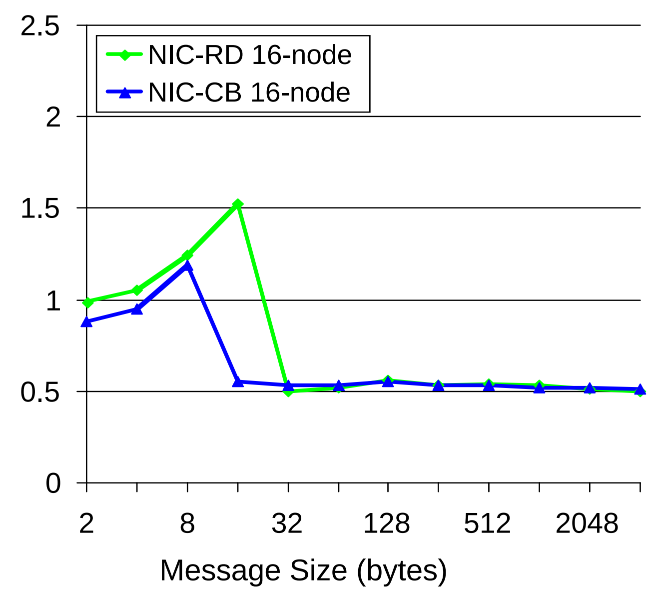
- For small messages, NIC-RD scales the best for being able to aggregate packets
- For large messages, NIC-CB performs the best

CPU Utilization

Host-Based



NIC-Based



- CPU utilization for host-based operation is high for the need to participate in polling and forwarding of intermediate messages
- With NIC-based all-to-all broadcast, it is low since host CPU only needs to post and later check for the completion of all-to-all broadcast operation

Presentation Outline

- Motivation
- Design Challenges and Implementation
- Performance Evaluation
- Conclusions and Future Work

Conclusions

- Characterized the challenges to efficient NIC-based all-to-all broadcast operations
- Proposed and designed two algorithms to overcome the constraints of NIC-based operations
- Implemented scalable and high-performance NIC-based all-to-all broadcast and have it integrated into a NIC-based collective protocol over Myrinet/GM

Future Work

- Evaluate the scalability of NIC-based all-to-all broadcast on large-scale systems
- Exploit the benefits of NIC-based All-to-All broadcast and the NIC-based collective package to applications or higher communication libraries

More Information

NBCL home page

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

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

Latest Results