Swapping to Remote Memory over InfiniBand: An Approach using a High Performance Network Block Device

Shuang Liang, Ranjit Noronha, D.K. Panda

Department of Computer Science and Engineering
The Ohio State University
Email: {liangs,noronha,panda}@cse.ohio-state.edu

Presentation Outline

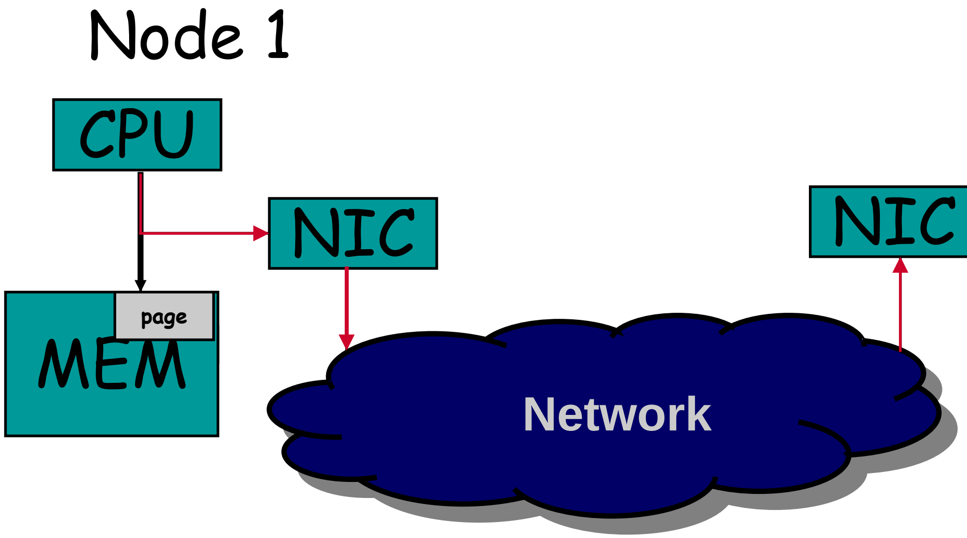
- Introduction
- Problem Statement
- Design Issues
- Performance Evaluation
- Conclusions and Future Work

Application Trends

- Applications are becoming increasingly data intensive with high memory demand
  - Larger working set for a single application, such as data warehouse application, scientific simulation, etc.
  - Memory resources in a single node of a cluster system may not be able to accommodate the working set in memory, while some other node may host plenty of memory unused

Utilizing Remote Memory

- Can we utilize those remote memory to improve the applications performance?



Motivation

- Emergence of commodity high performance network such as InfiniBand
  - Offloaded transport protocol stack
  - User-level communication bypass OS
  - Low latency
  - High bandwidth comparable to local *memcpy* performance
  - Novel hardware features such as Remote Direct Memory Access (RDMA) with minimal host overhead
- Can we utilize these features to boost sequential data intensive applications? And how?

Approaches

- Global memory management [Feeley95]
  - Close integration with virtual memory management
  - Implementation Complexity and poor portability
- User level run-time libraries [Koussih95]
  - Application aware interface
  - Additional management layer in user space
- Remote paging[Markatos96]
  - Flexible
  - Moderate implementation effort

Presentation Outline

- Background
- Problem Statement
- Design Issues
- Performance Evaluation
- Conclusions and Future Work

Problem Statement

- Enable InfiniBand cluster to take advantage of remote memory by remote paging
  - Enhance the local memory hierarchy performance
  - Deliver high performance
  - Enable application to benefit transparently
- Evaluate the network performance impact
  - Comparisons of remote paging with GigE, IPoIB and InfiniBand native communication

Presentation Outline

- Background
- Problem Statement
- Design Issues
- Performance Evaluation
- Conclusion and Future Work

Design Choices

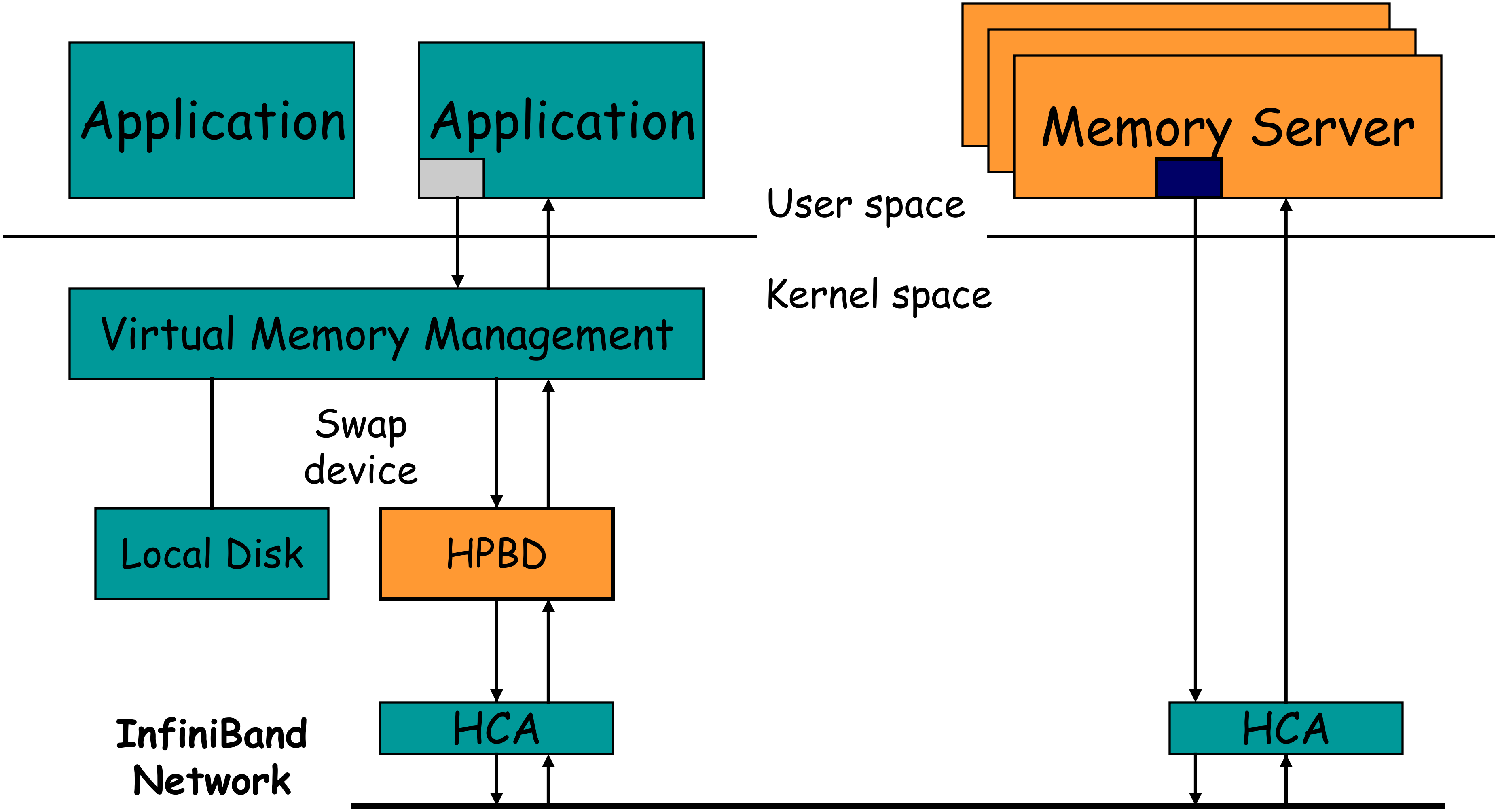
- Kernel Level Design
  - Pros:
    - Transparency to applications
    - Beneficial to processes in the system
    - Take advantage of virtual memory system management for page management
  - Cons:
    - Dependency on OS
    - Not easy to debug

- User Level Design
  - Pros:
    - Portable across different OSes
    - Easier to debug
  - Cons
    - Not completely transparent to application
    - Beneficial only to application using the user-level library
    - High overhead with user-level signal handling

Network Block Device

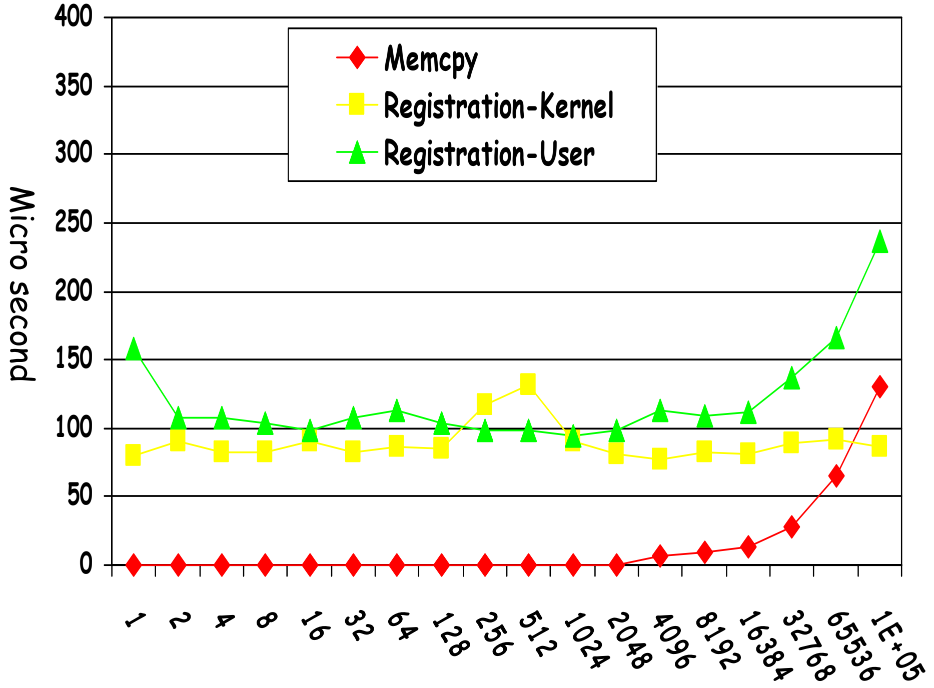
- A software mechanism to utilize remote block based resources over network
  - Examples: NBD, ENBD, DRBD, GNBD, etc.
  - Often used to export remote disk resources to provide storage, such as RAID device, mirror device, etc.
- Use Ramdisk based Network Block Device as swapping device
  - Seamless integration with VM for remote paging
  - NBD — a TCP implementation of Network Block Device within default kernel source tree can be used for comparison study
  - An InfiniBand based Network Block Device needs to designed

Architecture of the remote paging system



Design Issues

- Memory registration and buffer management
  - Registration is a costly operation compared with *memcpy* for small buffers
  - Pre-registration out of the critical path needs registration for all memory pages
  - Paging messages are upper bounded by 128KB in Linux



Size in Bytes

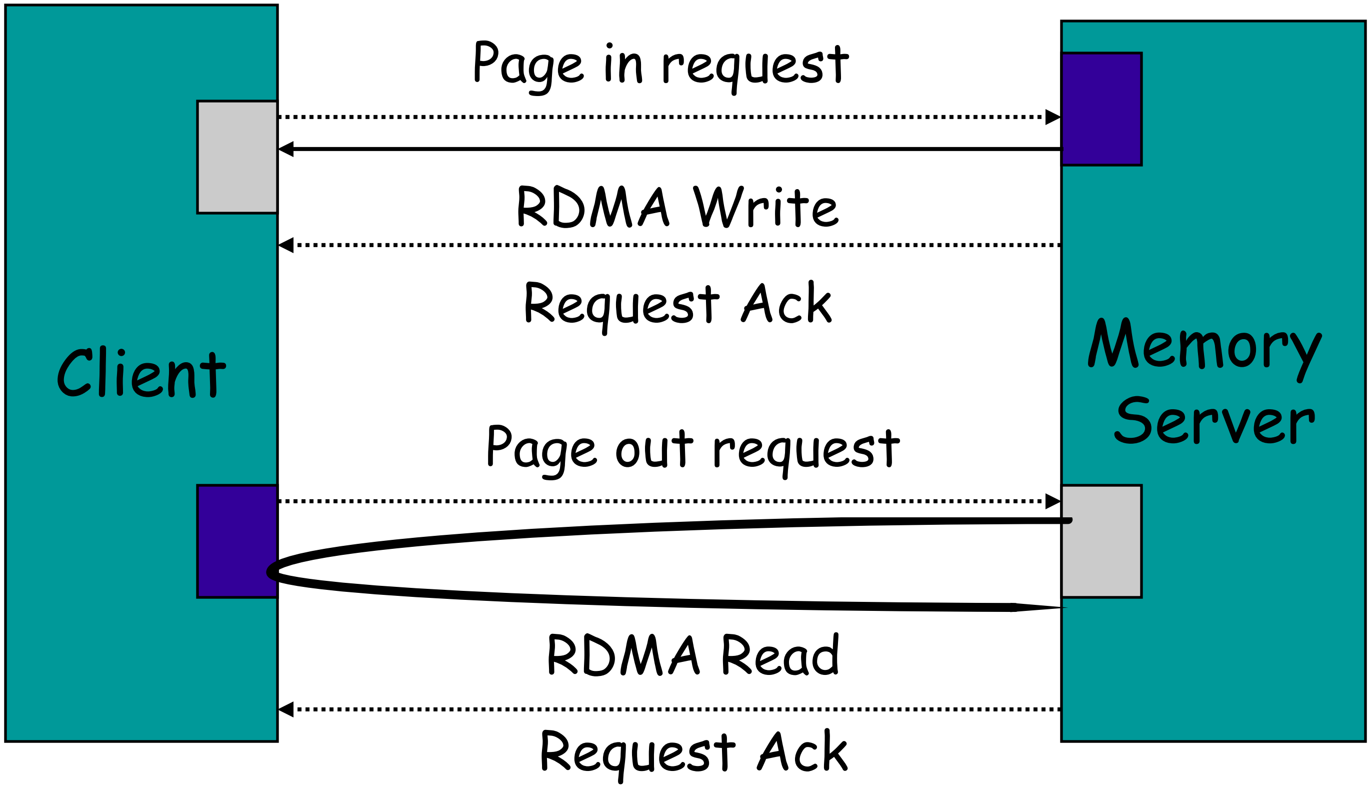
Memory copy is more than 12 times faster than memory registration for one page

Design Issues (cont'd)

- Dealing with message completions
  - Polling based synchronous completion wastes CPU cycles
  - None preemptive in kernel mode
  - InfiniBand supports event based completion by registering asynchronous event handler
- Thread safety
  - There could be multiple instances of the driver running, mutual exclusion is needed for shared data structures
- Reliability issues

Our Design

- RDMA based server design

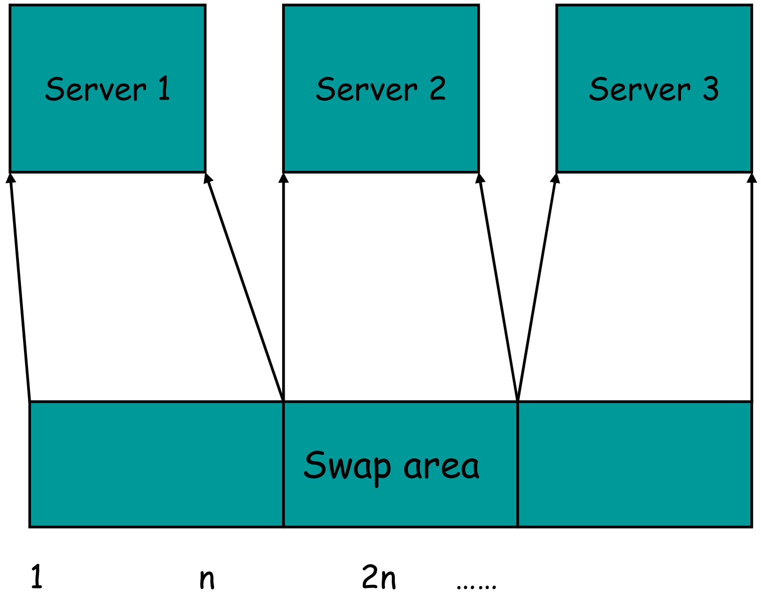


Our Design (cont'd)

- Registered buffer pool management
  - Use pre-register a buffer pool for page copy before communication
- Hybrid completion handling
  - Register an event handler with InfiniBand transport
  - Both client and server block, when there is no traffic
  - Use polling scheme for bursty incoming requests

Our Design (cont'd)

- Reliable communication
  - Using RC services
- Flow control
  - Use credit based flow control
- Multiple server support
  - Distribute block across multiple servers in linear mode



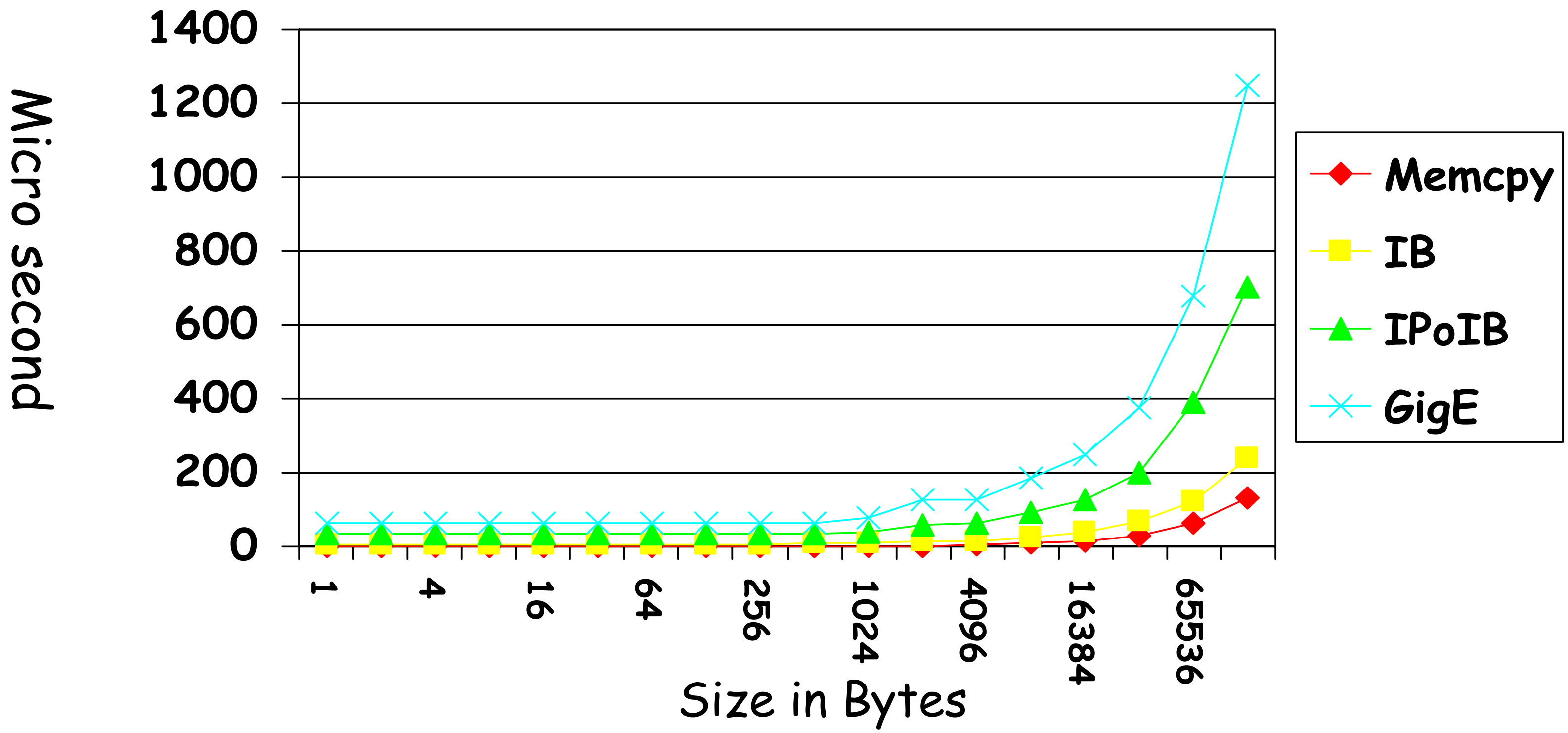
Presentation Outline

- Background
- Problem Statement
- Design Issues
- Performance Evaluation
- Conclusions and Future Work

Experiment Setup

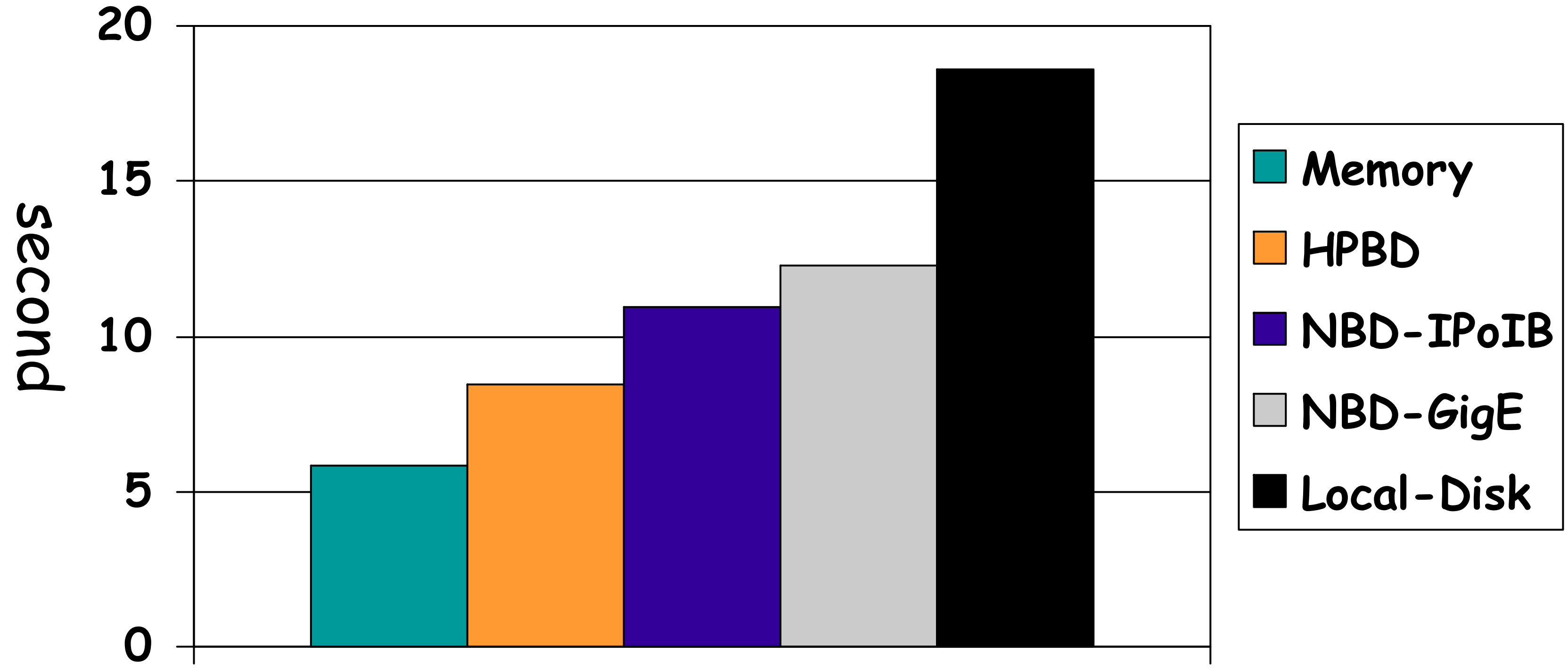
- Xeon 2.66GHZ Cluster with 2G DDR Memory; 40GB ST340014A Hard disk; InfiniBand Mellanox MT23108 HCA
- Memory size configuration:
  - 2G for local memory test scenario
  - 512M for swapping scenario
- Swapping area setup
  - Use Ram disk on memory server as swap area

Latency Comparison



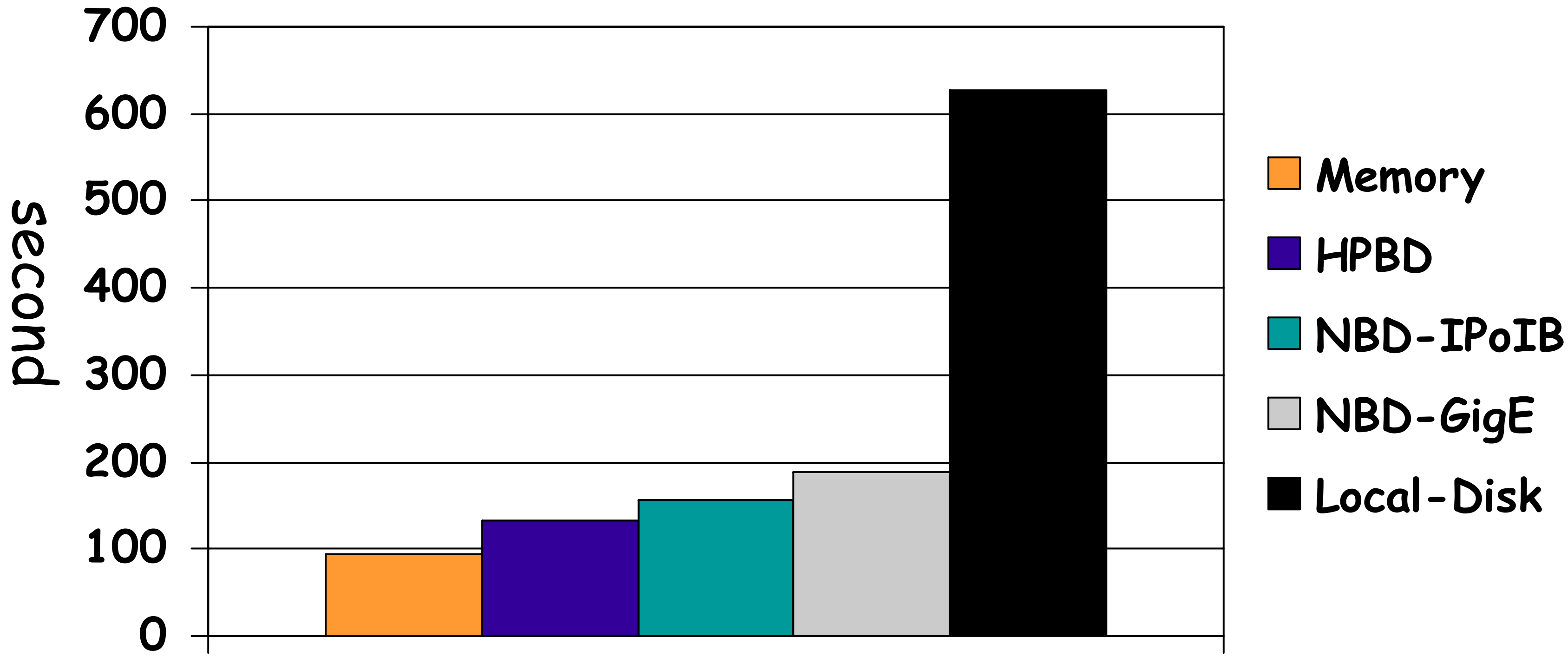
InfiniBand native communication latency for one page is 4 times faster than IPoIB and 8 times faster than GigE and 2.4 times slower than memcpy

Micro-benchmark: Execution Time



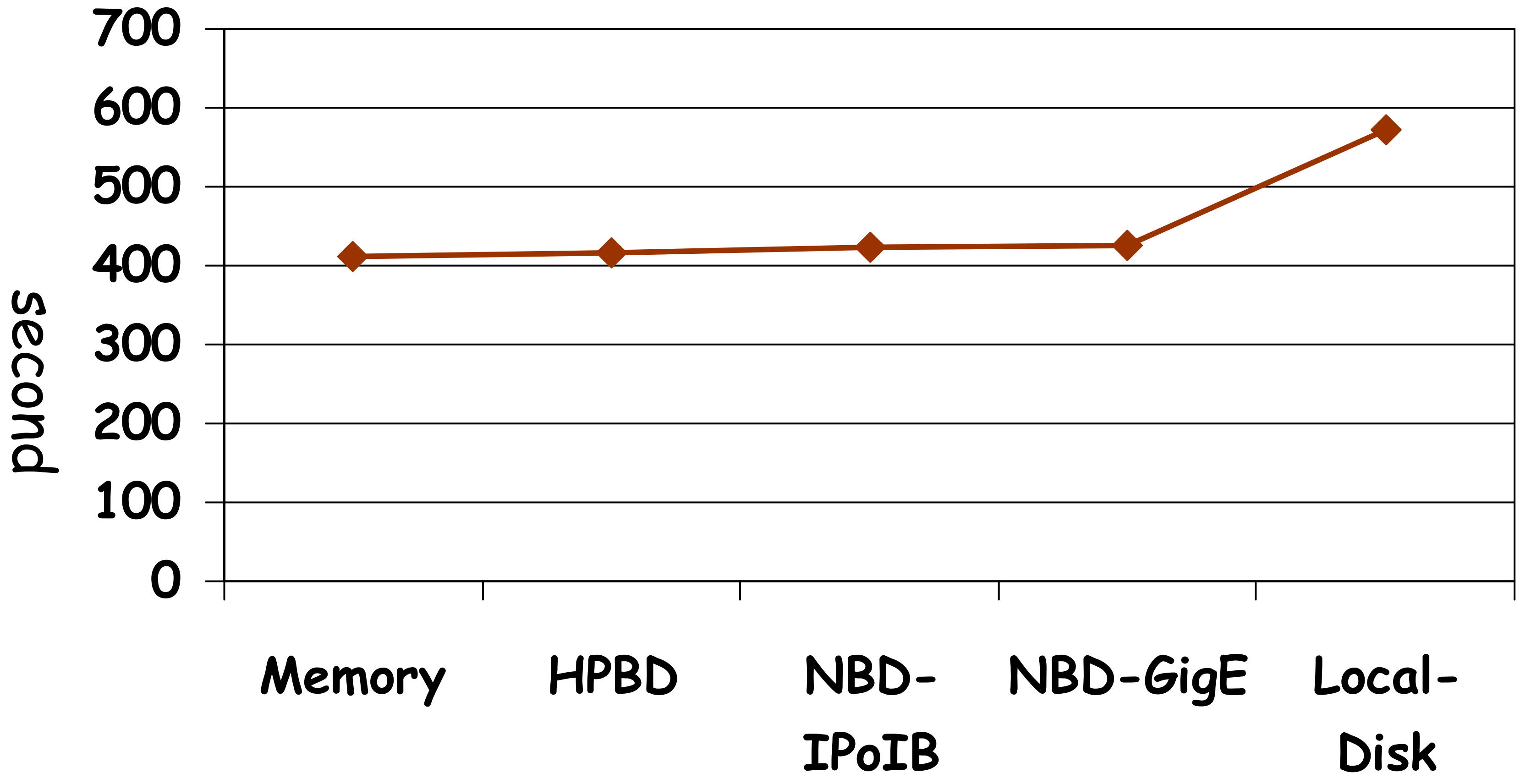
Network Overhead is approximately 30% for IPoIB. Using server based RDMA further improves the performance for HPBD

Quicksort - Execution time



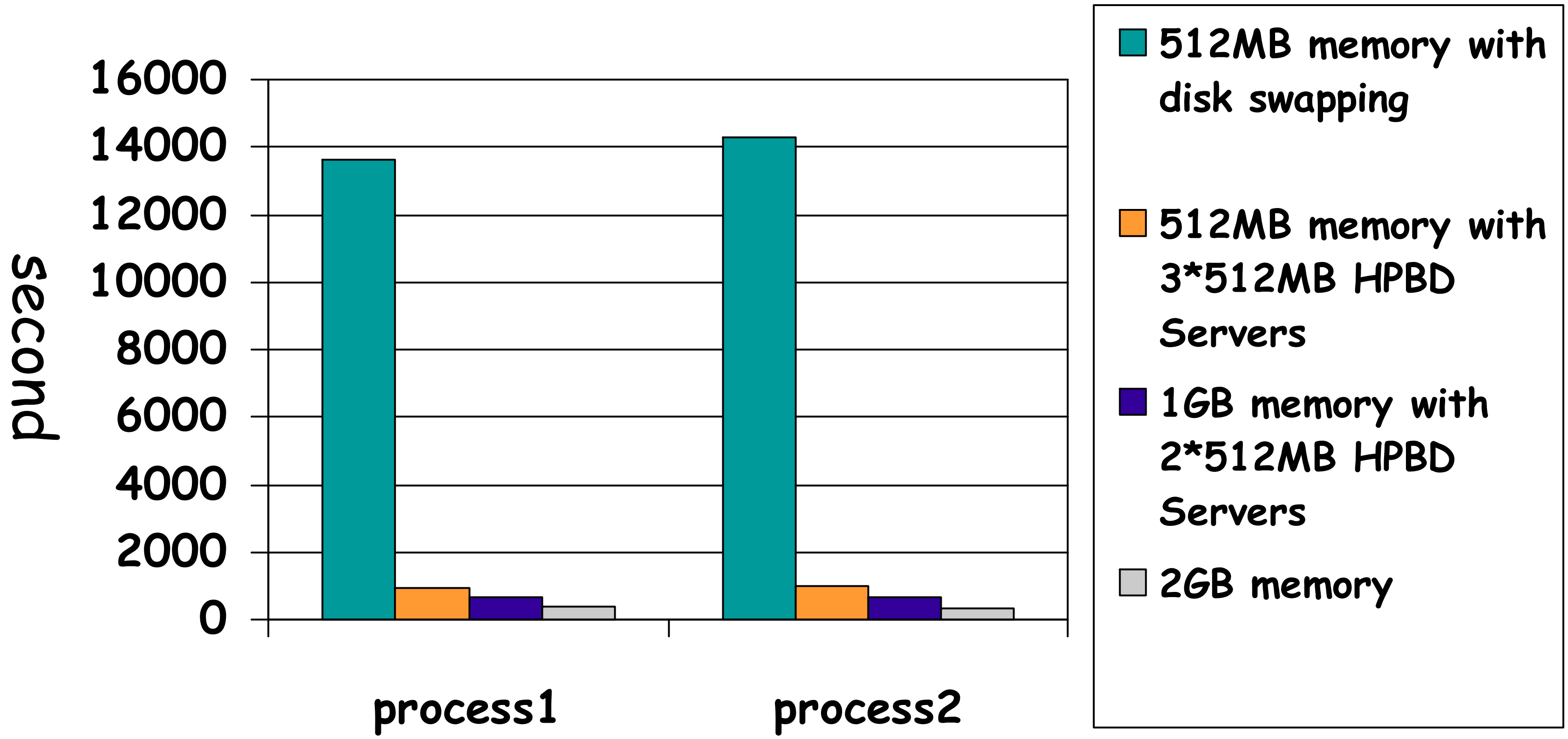
HPBD is 1.4 times slower than enough local memory and 4.7 times faster than swapping to disk

Barnes - Execution time



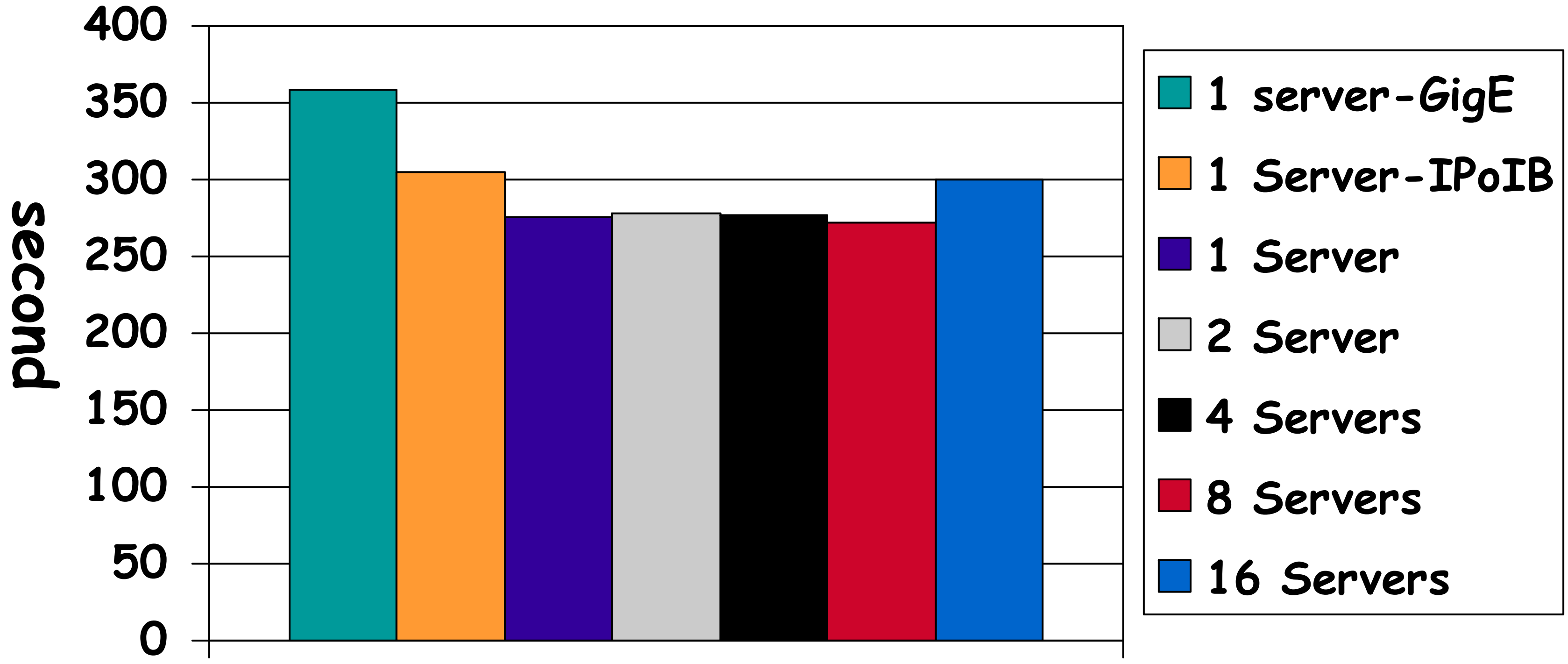
For slightly oversized working set, HPBD is still 1.4 times faster than swapping to disk

Two processes of Quicksort



Concurrent instances of quicksort run up to 21 times faster than swapping to disk

Quicksort with multiple servers



Maintaining multiple connections does not degrade performance up to 12 servers

Presentation Outline

- Background
- Problem Statement
- Design Issues
- Performance Evaluation
- Conclusions and Future Work

Conclusions

- Remote paging is an efficient way to enable sequential applications to take advantage of remote memory
- Using InfiniBand for remote paging can improve the performance, compared with GigE and IPoIB. And it is comparable to system with enough local memory
- As network speed increase, host overhead becomes more critical for further performance improvement

Future Work

- Achieve zero copy along the communication path to reduce host overhead along the critical path
- Dynamic management of idle cluster memory for swap area allocation

Acknowledgements

Our research is supported by the following organizations

- Current Funding support by

Office of Science U.S. DEPARTMENT OF ENERGY

NSF

intel.

Mellanox Technologies

Sun microsystems

- Current Equipment donations by

Mellanox Technologies

intel.

AMD

Sun microsystems

AMMASSO

Microway Technology you can count on

PathScale

SilverStorm

Thank You!

{liangs, noronha, panda}@cse.ohio-state.edu

Network-Based Computing Laboratory

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