Building Multirail Infiniband Clusters: MPI-Level Designs and Performance Evaluation

Mingzhe Li
Motivation

- Network bandwidth is a performance bottleneck for cluster computing. Especially for clusters built with SMP machines.

- Multirail network is an efficient way to alleviate this problem.
Background Information about Multirail

- Basic idea: multiple independent networks (rails) to connect nodes in a cluster.

- Two ways of using multirail:
  - Multiplexing:
    Multiple messages are combines together and send as one message through one rail.
  - Striping:
    Message is striped and sent as multiple message through multiple rails.
Multirail Network Types

- **Multiple HCAs**
  Multiple HCAs in each node and connect them to IB switch.

- **Multiple Ports**
  One HCA has multiple ports.

- **Single Port with LID Mask Control**
  Set up different paths between two ports on two nodes.
Multirail MPI Design

Figure 2. Basic Architecture of Multirail MPI Design
Virtual Subchannel Abstraction

- MPI Processes use virtual channel to communicate.
- Virtual channel contains information about different sub channels to provide a unified solution.

Figure 3. Virtual Subchannel Abstraction
Integration with MPI

- Multiplexing:
  RDMA, send/receive
  Scheduling policy: round robin

- Striping:
  RDMA
  Scheduling policy: adaptive striping
Adaptive Striping

- Goal: Transmission of each stripe finishes at about the same time.

- Three steps:
  - Set fixed weight at initialization time
  - Monitor progress of different sub channel
    \[ T_1 = \text{completion notification time} - \text{stripe start time} \]
  - Exploit feedback. Adjust weight when all stripes finish
Performance Evaluation

Figure 5. MPI Latency (UP mode)

Figure 6. MPI Bandwidth (Small Messages, UP mode)

Figure 7. MPI Bandwidth (UP mode)
Improving Application Performance and Predictability using Multiple Virtual Lanes in Modern Multi-core Infiniband Clusters
Motivation

- Network congestion is an important factor affecting the performance of clusters.
- Ping-pang benchmark to test the performance degradation:

![Graphs showing packet arrival times and frequency]

Fig. 2. Distribution of Inter Transmission Time of 99.999% of Packets for (a) One Pair of Processes and, (b) Eight Pairs of Processes
IBA QoS

- Components of IBA QoS mechanism:
  Virtual Lane, VL arbitration, link level flow control

- Service Level QoS:
  - Service Level to Virtual Lane mapping table
  - Virtual Lane arbitration table: defines two priority levels. High priority entries will be processed first.
  - In each priority level, weighted round-robin scheme is used.
Design

- Various ways buffer can be split.
- In this paper, the design is based on private virtual lane buffer.
Schemes with VL

- Traffic distribution
  Distribute the traffic of entire job evenly across all available virtual lanes.

- Traffic segregation
  Isolate applications as much as possible from being adversely impacted by other applications.
Native IB Results

Fig. 7. Impact of using Multiple Virtual Lanes on Native IB Latency for (a) 2 KB Message Size and, (b) 4 KB Message Size

Fig. 8. Impact of Traffic Distribution on Native IB Performance for 2 KB Message Size with (a) One VL and, (b) Eight VL’s
Collective Result

Fig. 14. Impact of Traffic Distribution on Performance of 64 Process Alltoall for 4KB Message Size with (a) One VL and, (b) Eight VL’s
Application Performance

![Bar chart showing normalized time for total time and time in AlltoAll for CPMD Application Parameters with One VL and Eight VL's.]

Fig. 18. Impact of Traffic Distribution on Performance of 64 Process CPMD Application
Application-transparent checkpoint/restart for MPI programs over Infiniband
Motivation

- Failure rate on ultra-scale clusters increase along with their augmented number of components.
- Faults wastes a large amount of computing resources because all the computation that has already been accomplished is lost.
Rollback Recovery Protocols:

- **Message logging**
  Cons: Impose overhead in the environment of high bandwidth interconnects

- **Solely uncoordinated checkpointing**
  Cons: Domino effect: dependency between processes may make all processes roll back to the initial state

- **Coordinated checkpointing** (this paper’s approach)
Levels of Checkpoint

- Application level checkpointing
  Pros: gain advantages of efficiency with the assistance from user application.
  Cons: source code of application needs to be tailored.

- System level checkpointing (this paper’s approach)
  Cons: more overhead than application level
  Pros: no need to modify application
Challenges

- Parallel process over IB communication via an OS-bypass user-level protocol.
- Context of network connection is available on in network adapter.
- Some network context is cached on the remote node.
Figure 1. Checkpoint/Restart Framework
C/R Procedure

- **Initial Synchronization Phase:**
  C/R controller wake up and lock communication
- **Pre-checkpoint coordination phase:**
  Suspend communication channel
- **Local checkpointing phase:**
  Save state of local MPI process and suspended channel
- **Post-checkpoint coordination phase:**
  Reactivate communication channel.
Suspension/Reactivation IB

- **Suspension**
  Drain all the in-transit message (network level rather MPI messages). Achieved by Flag, process the control message and buffer it.

- **Reactivation**
  Update local communication channel, send control messages to update the other side of the channel
Performance

Figure 4. Overall Time for Checkpointing/Restarting NAS

Figure 5. Coordination Time for Checkpointing/Restarting NAS
Performance Impact