Multicast in Fat-Tree-Based InfiniBand Networks

Jiazheng Zhou, Xuan-Yi Lin, Chun-Hsien Wu, and Yeh-Ching Chung
Department of Computer Science
National Tsing-Hua University, Hsinchu, Taiwan 30013, ROC
{jzzhou, xylin, chwu, ychung}@cs.nthu.edu.tw

Abstract

The multicast operation is a very commonly used operation in parallel applications. With the hardware supported multicast of the InfiniBand Architecture (IBA), we propose a cyclic multicast scheme for fat-tree-based (m-port n-tree) InfiniBand networks. The basic concept of the proposed cyclic multicast scheme is to find the union sets of the output ports of switches in the paths between the source processing node and each destination processing node in a multicast group. Based on the union sets and the path selection scheme, the forwarding table for a given multicast group can be constructed. We implement the proposed multicast scheme along with the OpenSM multicast scheme and the unicast scheme on an m-port n-tree InfiniBand network simulator. The simulation results show that the proposed multicast scheme outperforms the unicast scheme for all simulated cases. For many-to-many and all-to-many cases, the cyclic multicast scheme outperforms the OpenSM multicast scheme. For many-to-all case, the performance of the cyclic multicast scheme is a little better than that of the OpenSM multicast scheme.

1. Introduction

Interconnection networks in cluster systems have great impact on the performance of communication-bounded applications. The InfiniBand Architecture (IBA) [2] is a new industry-standard architecture for server I/O and inter-server communication. The IBA enables high-speed, low-latency communication between connected devices and it is suitable for the interconnection network of a cluster system.

The multicast operation [1,7] is a very commonly used operation in cluster systems. Since the InfiniBand Architecture supports hardware multicast, one can take advantage of this feature to speedup the multicast operation. OpenSM [6] is an implementation of subnet manager. In OpenSM, it implements a hardware supported multicast scheme by using a spanning tree approach to construct the multicast paths for a given multicast group. However, its performance may not be satisfied since it does not take the characteristics of a network topology into account.

In this paper, we focus on the fat-tree topology [3-4,8] used in most scalable cluster systems nowadays. We propose a *cyclic* multicast scheme for the *m*-port *n*-tree (a fat-tree) InfiniBand networks [5] based on the hardware supported multicast feature of the IBA and the characteristics of *m*-port *n*-tree fat-trees. To evaluate the proposed method, we implement the cyclic multicast scheme, the OpenSM multicast scheme, and a unicast scheme on an *m*-port *n*-tree InfiniBand network simulator that is written in Java. The simulation results show that the proposed cyclic multicast scheme outperforms the unicast schemes for all test cases. The lager the message size, the number of multicast source nodes, and the size of the multicast group, the better speedup can be expected from the proposed multicast schemes.

The rest of this paper is organized as follows. Section 2 will introduce the fat-tree-based InfiniBand networks. The proposed multicast schemes will be described in Section 3. Section 4 will give the simulation results for the proposed multicast schemes. The conclusions will be given in Section 5.

2. Preliminaries

2.1. InfiniBand Architecture (IBA)

The InfiniBand network is a packet-switching network. Since the mapping between destination local identifier (DLID) and output port is one-to-one, in order to support multiple paths, the IBA defines an LID Mask Control (LMC) value that can be assigned to each endport. The LMC is a 3-bit field that represents 2^{LMC} paths (maximum of 128 paths).

The IBA also supports hardware multicast. In the IBA, each multicast group is assigned a multicast LID and a global identifier (GID) by the subnet manager. The subnet manager will setup the forwarding table of

Proceedings of the 2005 Fourth IEEE International Symposium on Network Computing and Applications (NCA'05)
0-7695-2326-9/05 $20.00 © 2005 IEEE

each switch for each multicast group according its LID and GID. To perform a multicast operation in an InfiniBand network, the source processing node uses the multicast LID and the GID of a multicast group to send packets. When a switch receives a multicast packet, it replicates the packet and forwards the packet to the corresponding output ports according to its forwarding table.

2.2. The m-Port n-Tree InfiniBand Networks

In [5], we have proposed an *m*-port *n*-tree InfiniBand network *IBFT*(*m*, *n*). It has the following characteristics:

1. The height of *IBFT*(*m*, *n*) is $n + 1$.
2. *IBFT*(*m*, *n*) consists of $2(m/2)^n$ processing nodes and $(2n-1)(m/2)^{n-1}$ InfiniBand switches.
3. Each switch has *m* ports.

An example is shown in Figure 1.

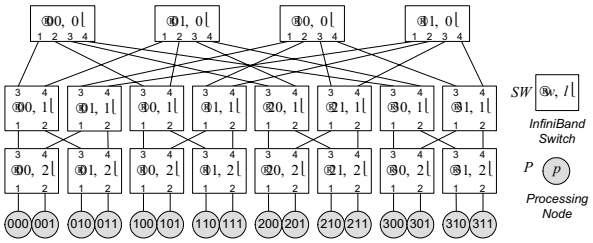


Figure 1. An example of a 4-port 3-tree InfiniBand network.

3. The Proposed Multicast Scheme

The proposed multicast scheme consists of three sub-schemes, the processing node addressing scheme, the path selection scheme, and the forwarding table assignment scheme. We use the four definitions in [5], the greatest common prefix, the least common ancestors, the greatest common prefix group, and the rank of processing node.

3.1. The Processing Node Addressing Scheme

Given an *m*-port *n*-tree InfiniBand network *IBFT*(*m*, *n*), in the multicast scheme, every processing node in *IBFT*(*m*, *n*) is assigned a set of LIDs. The set of LIDs assigned to each processing node is formed by the combination of one base LID and a LID Mask Control value *LMC*, where $LMC = \log_2(m/2)^{n-1}$. For processing node $P(p \mid p_0 p_1 \cdots p_{n-1})$ in *IBFT*(*m*, *n*), the set of LIDs assigned to *P*(*p*), denoted by *LIDset*(*P*(*p*)), is {*BaseLID*(*P*(*p*)), *BaseLID*(*P*(*p*)) + 1, …, *BaseLID*(*P*(*p*)) + $(2^{LMC}-1)$}, where $BaseLID(P(p)) = 2^{LMC}\left(\sum_{i=0}^{n-1} p_i (m/2)^{n-(i+1)}\right) + 1$ is the base LID of *P*(*p*). There are 2^{LMC} LIDs in *LIDset*(*P*(*p*)), which indicates that there are maximal 2^{LMC} paths between any pair of processing nodes.

3.2. The Path Selection Scheme

We propose a *cyclic* path selection scheme according to the cyclic grouping policy to take the advantage of multiple LIDs of a processing node such that the duplication of a packet will not occur in the ascending phase. The grouping policy is to decide what processing nodes are in the same group for a given destination processing node *P*(*p*). For the processing nodes in the same group, they will send messages to the destination processing node *P*(*p*) by choosing the same LID of *P*(*p*).

Given an *m*-port *n*-tree InfiniBand network *IBFT*(*m*, *n*), for a destination processing node $P(p \mid p_0 p_1 \cdots p_{n-1})$, source processing nodes $P(s_1)$ and $P(s_2)$ are in the same cyclic group $CG(P(p), l, y)$ if the following two rules are satisfied.

Rule 1: The level *l* of the least common ancestors *lca*(*P*(*p*), *P*(s_1)) is the same as that of *lca*(*P*(*p*), *P*(s_2)).

Rule 2: $P(s_1)$ and $P(s_2)$ have the same common suffix *y* and $|y| = n-1-l-1$.

The cyclic path selection scheme is performed as follows. For a destination processing node $P(p \mid p_0 p_1 \cdots p_{n-1})$ and a source processing node $P(p' \mid p'_0 p'_1 \cdots p'_{n-1})$ in $CG(P(p), l, y)$, when $P(p')$ wants to send messages to *P*(*p*), it will select $BaseLID(P(p)) + rank(gcpg(p'_0 p'_1 \cdots p'_l, l+1), P(p'))$ as the LID of *P*(*p*).

Figure 2 shows an example of the cyclic path selection scheme for a 4-port 3-tree InfiniBand network *IBFT*(4, 3). Given an destination processing node *P*(200), we can divide the source processing nodes into cyclic groups $CG(P(200), 0, 00) = \{P(000), P(100), P(300)\}$, $CG(P(200), 0, 01) = \{P(001), P(101), P(301)\}$, $CG(P(200), 0, 10) = \{P(010), P(110), P(310)\}$, $CG(P(200), 0, 11) = \{P(011), P(111), P(311)\}$, $CG(P(200), 1, 0) = \{P(210)\}$, $CG(P(200), 1, 1) = \{P(211)\}$, and $CG(P(200), 2, \kappa) = \{P(201)\}$ based on the cyclic grouping policy, where κ is a null string. Assume that there are four source processing nodes *P*(000), *P*(001), *P*(010), and *P*(011) want to send messages to the destination processing node *P*(200). Since the four source processing nodes are in different groups of *CG*(*P*(200), 0), they will choose the different LIDs 33, 34, 35, and 36 (33 + 0, 33 + 1, 33 + 2, and 33

Proceedings of the 2005 Fourth IEEE International Symposium on Network Computing and Applications (NCA'05)
0-7695-2326-9/05 $20.00 © 2005 IEEE

+ 3) of the destination processing node $P(200)$ and send messages through paths Q, R, S, and T, respectively.

According to the path selection scheme, the duplication of packets can be avoided in the ascending phase when a processing node sends packets to different destination processing nodes. An example is shown in Figure 3. In Figure 3, the source processing node $P(000)$ sends messages to $P(200)$, $P(201)$, $P(210)$, and $P(211)$ through routes Q, R, S, and T, respectively. From Figure 3, we can see that all routes take the same path in the ascending phase.

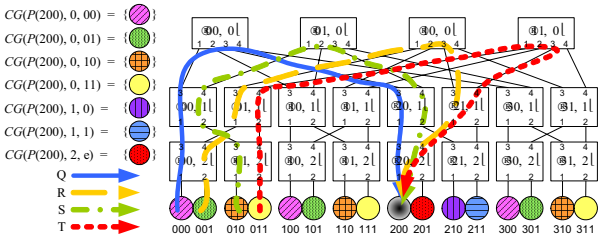


Figure 2. The cyclic path selection scheme.

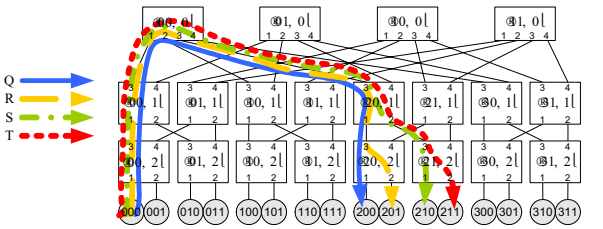


Figure 3. An example of one-to-many multicast.

3.3. The Forwarding Table Assignment Scheme

The forwarding table assignment consists of two phases: the one-to-one forwarding table assignment and the multicast forwarding table assignment based on union operation.

3.3.1. The One-to-one Forwarding Table Assignment

The one-to-one forwarding table assignment is described in section 4.3 of [5].

3.3.2. The Multicast Forwarding Table Assignment Based on Union Operations

After the one-to-one forwarding table assignment is performed, we can setup the multicast forwarding table for a given source processing node and a multicast group based on union operations. Let $P(p \mid p_0 p_1 \cdots p_{n-1})$ be a source processing node and $lid = \{lid_1, lid_2, ..., lid_t \mid t \le 2(m/2)^n\}$ be the DLID of a multicast group, where $\{lid_1, lid_2, ..., lid_t \mid t \le 2(m/2)^n\}$ is the set of LIDs of destination processing nodes in a multicast group. For each switch $SW\langle w, l\rangle$, based on Equations (1) and (2), we can determine the output port of a packet whose DLID is lid_1, lid_2, …, and lid_t as $SW\{w,l\}_{k_1}$, $SW\{w,l\}_{k_2}$, … , and $SW\{w,l\}_{k_t}$, respectively. It means that when a packet whose DLID is lid_1, lid_2, …, and lid_t arrives in switch $SW\langle w, l\rangle$, it will be forwarded to port $SW\{w,l\}_{k_1}$, $SW\{w,l\}_{k_2}$, … , and $SW\{w,l\}_{k_t}$, respectively. Since an InfiniBand switch can duplicate a packet to different output ports and the path selection schemes given in Section 3.2 will prevent the packet from being duplicated in the ascending phase, the output ports of a multicast packet $\{lid_1, lid_2, ..., lid_t \mid t \le 2(m/2)^n\}$ can be set as the union of $SW\{w,l\}_{k_1}$, $SW\{w,l\}_{k_2}$, … , and $SW\{w,l\}_{k_t}$ in switch $SW\langle w, l\rangle$.

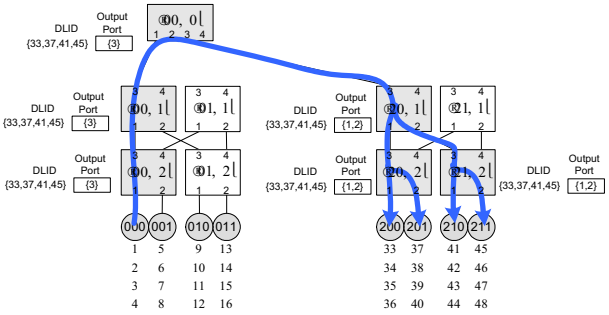


Figure 4. An example of multicast forwarding table setup.

An example is shown in Figure 4. In Figure 4, assume that processing node $P(000)$ wants to send multicast packets to processing nodes $P(200)$, $P(201)$, $P(210)$ and $P(211)$. The *lid* set of the multicast group is {33, 37, 41, 45}. From Figure 5, we can determine that its output ports in switch $SW\langle 00, 2\rangle$ = {3}, $SW\langle 00, 1\rangle$ = {3}, $SW\langle 00, 0\rangle$ = {3}, $SW\langle 20, 1\rangle$ = {1, 2}, $SW\langle 20, 2\rangle$ = {1, 2}, and $SW\langle 21, 2\rangle$ = {1, 2}, respectively. The multicast operation can be performed correctly.

4. Performance Evaluation

To evaluate the performance of the proposed multicast scheme, we design an m-port n-tree InfiniBand network simulator by using Java. Three schemes, the proposed cyclic multicast scheme, the OpenSM multicast scheme, and the unicast scheme were simulated for performance evaluation.

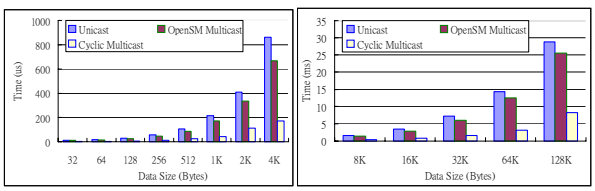
An 8-port 3-tree InfiniBand network is simulated. The network contains 80 switches and 128 processing nodes. Here we only show the simulation results of 40%-to-40% multicast in Figure 5 to Figure 7. Since there are more than one source processing nodes send messages to the destination processing nodes, the traffic congestion did occur. From the simulation results, we can see that the multicast schemes outperform the unicast scheme. Moreover, our cyclic multicast scheme outperforms the OpenSM multicast scheme when the destination group size is small (10% and 40%). This is because the OpenSM multicast scheme only builds one multicast tree, while our scheme builds more multicast trees to take the advantages of available bandwidth of fat-tree topology. When the destination group size is large (100%), the performance of our scheme is a little better than that of the OpenSM multicast scheme since the serious traffic congestion in the descending phase.

5. Conclusions

In this paper, we propose a hardware supported multicast scheme for the fat-tree-based InfiniBand networks. The simulation results show that the proposed cyclic multicast scheme can speed up the execution of multicast operations. From the simulations results, we have the following remarks:

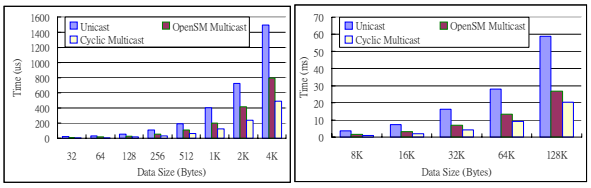
Remark 1: We observe that the proposed multicast scheme outperforms the unicast scheme for all simulated cases. This result indicates that the hardware supported multicast of the IBA can help to speedup the execution of multicast operations.

Remark 2: Comparing to the OpenSM multicast scheme, for one-to-many case, the performance of the cyclic multicast scheme is the same as that of the OpenSM multicast method. For many-to-many and all-to-many cases, the cyclic multicast scheme outperforms the OpenSM multicast method. For many-to-all case, the performance of the cyclic multicast scheme is a little better than that of the OpenSM multicast method.



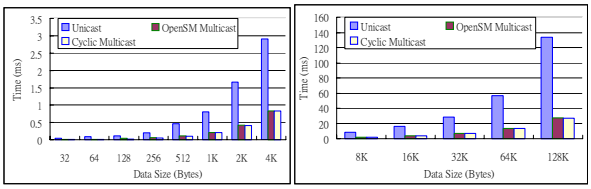
(a) Small size (b) Large size

Figure 5. 40%-to-10% multicast.



(a) Small size (b) Large size

Figure 6. 40%-to-40% multicast.



(a) Small size (b) Large size

Figure 7: 40%-to-all multicast.

Acknowledgments

The work in this paper was partially supported by National Science Council and Ministry of Economic Affairs of the Republic of China under contract NSC-93-2213-E-007-100, NSC-93-2752-E-001-004-PAE and 94-EC-17-A-01-S1-038.

References

[1] J. Duato, S. Yalamanchili, and L. Ni, *Interconnection Networks - An Engineering Approach*, IEEE CS Press, 1997.

[2] InfiniBand™ Trade Association, *InfiniBand™ Architecture Specification Volume 1, Release 1.2*, October 2004.

[3] S. Kumar and L. V. Kale, "Scaling Collective Multicast on Fat-Tree Networks," To appear in *International Conference on Parallel and Distributed Systems*, 2004.

[4] C. E. Leiserson, "Fat-Trees: Universal Networks for Hardware-Efficient Supercomputing," *IEEE Transactions on Computers*, vol. 34, no. 10, October 1985, pp. 892-901.

[5] X. Y. Lin, Y. C. Chung, and T. Y. Huang, "A Multiple LID Routing Scheme for Fat-Tree-Based InfiniBand Networks," *Proceedings of IEEE International Parallel and Distributed Proceeding Symposiums* (CD-ROM), April 2004.

[6] Linux InfiniBand Project. http://infiniband.sourceforge.net.

[7] P. López, J. Flich, and J. Duato, "Deadlock-Free Routing in InfiniBand™ through Destination Renaming," in *Proceedings of the International Conference on Parallel Processing*, *ICPP '01*, Sept. 2001, pp. 427-434.

[8] F. Petrini and M. Vanneschi, "k-ary n-trees: High Performance Networks for Massively Parallel Architectures," in Proceedings of the 11th International Parallel Processing Symposium, IPPS'97, April 1997, pp. 87-93.