Performance-Effective and Contention-Free Broadcasts on Irregular Network with Heterogeneous Workstations

Ching-Hsien Hsu*, Ming-Hsiung Tsai, Tai-Lung Chen+ and Kun-Ming Yu
Department of Computer Science and Information Engineering
+Institute of Engineering and Science
Chung Hua University, Hsinchu, 30012, Taiwan
**E-mail: chh@chu.edu.tw*

Abstract

With the advance of network and computer techniques, the development of scalable computing becomes a new trend. To integrate and utilize distributed and heterogeneous resources efficiently, message broadcasting is an important and crucial technique for such systems. In this paper, we present a Location Aware Broadcast Scheme (LABS) for performing broadcast on irregular heterogeneous network. The LABS introduces a new scheduling scheme that based on heterogeneity of workstation and network topology. Together with a binomial tree optimization technique, LABS can arrange communications in a contention free and shortest routing path manner. To evaluate the performance of LABS, we have implemented the proposed techniques along with other algorithms. The experimental results show that LABS has good performance in different circumstances. Especially, LABS has significant improvements when the environment is with high heterogeneity.

1. Introduction

Broadcasting is a common operation in various network applications. The important things for efficient broadcasting are to minimize communication latency and improve network utilization. Constructing a scheduling tree for broadcasting is the method used frequently, but it is an *NP-Complete* problem to get an optimal scheduling tree. Due to the advancement of equipments such as workstations and network devices, Heterogeneous Networks of Workstations (*HNOW*) becomes a widely accepted paradigm for distributed computing. On *HNOW*, workstations have various characteristics including CPU speed, memory size, and communication speed. These parameters are crucial factors that affect the performance of broadcasting.

In this paper, our intention is to construct a performance-effective scheduling tree for broadcasting on Switch-Based *HNOW*. Figure 1 depicts an example of Switch-Based *HNOW*. A rectangle represents a switch having eight ports and connecting to workstations or other switches. All links in Switch-Based *HNOW* are bidirectional between any two switches. It means that opposite directions of every link, which connects between any two switches, can be used simultaneously without link contention. The three symbols represent the workstations connected to switches; different symbols represent different speed types.

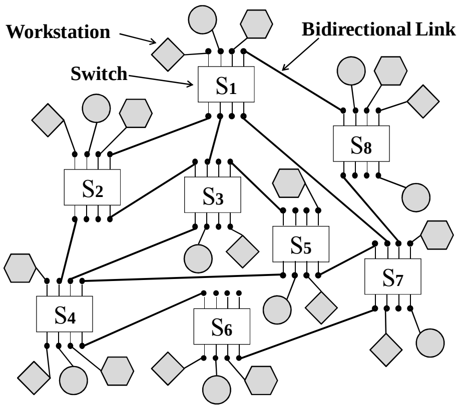


Figure 1. An example of Switch-Based *HNOW*

Amit Singhal *et al.* presented different approaches to deal with the multicast on Switch-Based *HNOW* that contains two speed types [11]. The algorithm provided an optimal scheduling tree constructed by dynamic programming, but it is only routing with two speed types. Lin *et al.* presented a scheme called *TWO-VBBS* [7] that combines two algorithms, one is network partition [8] and the other one is *VBBS* [6]. It makes broadcasting with more than two speed types and contention-free on Switch-Based *HNOW*. However, *TWO-VBBS* cannot ensure that any of the workstations sends the message to another in the shortest path and workstations with fastest speed type are always in the upper level of the scheduling tree. Hence, we aim to design a scheduling tree broadcast the messages between workstations in the shortest paths, and make workstations with fastest speed type are in the upper level of the scheduling tree.

The rest of this paper is organized as follows. Section 2 presents related works. Section 3 proposes cost model. Section 4 describes location oriented spanning tree construction. Section 5 expresses the scheduling tree for broadcasting. Section 6 shows performance evaluation. Section 7 presents conclusions.

2. Related Works

2.1 Wormhole Routing

The wormhole routing technology having low communication latency was presented in [10]. For receiving a packet completely in a workstation and then sending to the next workstation, wormhole routing operates by advancing the head of a packet directly from incoming to outgoing ports of the routing switch. In the wormhole routing, a packet is divided into a number of *flits* (flow control digits) for transmission. The size of a *flit* relies on system parameters, in particular, the port width of the switch. The header *flit* (or *flits*) of a packet is the key point to command the switch. Every time a workstation examines the header *flit*(*s*) of the message, it selects the next port of the switch and start forwarding *flits* down that port. As the header advances along the specified switch, the remaining flits follow in a pipeline fashion. There were several irregular Switch-Based networks, such as Myrinet [2], applied in the wormhole routing. We use this wormhole routing technology to make broadcasting on Switch-Based *HNOW* much lower communication latency in this paper.

2.2 The Binomial Tree

The importance of efficient broadcasting is to get the minimal number of communication steps, constructing a scheduling tree is the most way for broadcasting. McKinley *et al.* [9] used a binomial tree to implement the contention-free multicast in $\lceil log_2(\boldsymbol{n}+1)\rceil$ steps, where $\boldsymbol{n}$ is the number of receivers; it had been proved that it is the minimal number of communication steps for broadcasting with $\boldsymbol{n}$ receivers.

In [4], a fundamental policy for constructing a binomial tree is discussed. To perform a broadcast, a list of workstations is firstly given; the left most workstation in this list represents the source workstation. The source workstation firstly sends the message to the list center, the value of the list center is $\lceil \boldsymbol{n}/2\rceil$. The source workstation sends the message to another list center that is between the source workstation and first list center. Every list center, which has received the message as the source, sends the message to its list center. They do this operation repeatedly until all workstations in this list get the message. We design the binomial-like tree that is a little different with the binomial tree in section 5.1. Then, we prove that the communication step of the binomial-like tree is also $\lceil log_2(\boldsymbol{n}+1)\rceil$ in Lemma 2.

2.3 Contention-Free Scheduling

Link contention is an essential problem for broadcasting in the network. While it does not arrange all workstations properly in the scheduling tree, two different workstations may use the same direction link to send the messages at the same time. It makes more latency for broadcasting and decrease the performance of whole networks. *Up*/down** routing is commonly used in irregular wormhole routing networks as the basic routing to reduce link contention.

Kesavan *et al.* [3] and Libeskind-Hadas *et al.* [5] both use the ordered chain property to avoid contention: let the symbol $<_{od}$ denote such an ordering. If there exist four workstations *w*, *x*, *y*, *z*, and an ordered chain is $w <_{od} x <_{od} y <_{od} z$. The messages routing between workstations *w* and *x* will not contend for the messages routing in any links between workstations *y* and *z*, even *x* is equal to *y*. The *VBBS* algorithm [6] used the postorder doubling algorithm [5] containing *up*/down** routing and the ordered chain property as the basic routing to avoid contention.

3. Cost Model

On *HNOW*, the broadcast operation can be regarded as constitution of a number of P2P communications. Therefore, to estimate the cost of broadcasting a message on *HNOW*, we firstly formulate the cost of each individual P2P communication [1]. Then, the overall cost of the broadcast can be evaluated by merging these individual costs.

In general, a P2P communication consists of three costs, including the message sending cost at the sender, the message transmission cost through the underlying network and the message receiving cost at the receiver. The cost of a single P2P transmission on *HNOW* can be formulated as follows:

$$T_{ptp} = O_{send} + O_{trans} + O_{receive} \quad (1)$$

$$O_{send} = S_c^{Sender} + S_m^{Sender} \times m \quad (2)$$

$$O_{trans} = X_c + X_m \times m \quad (3)$$

$$O_{receive} = R_c^{Receiver} + R_m^{Receiver} \times m \quad (4)$$

Where
O_{send} is the message sending cost at the sender, in which S_c^{Sender} is the message startup cost; S_m^{Sender} is the communication latency per byte at the sender and *m* is the message size.

O_{trans} is the message transmission cost, in which X_c is the message startup cost at a switch and X_m is the cost of communication latency per byte through the network.

$O_{receive}$ is the message receiving cost at the receiver, in which $R_c^{Receiver}$ is the message startup cost; $R_m^{Receiver}$ is the communication latency per byte at the receiver.

Factors of S_c^{Sender}, X_c and $R_c^{Receiver}$ are constant costs of the sender, transmission and receiver, respectively. The factors S_m^{Sender}, X_m and $R_m^{Receiver}$ are dependent with the size of a message.

Considering the above P2P communication that performed on *HNOW* with multiple switches, the formulation of (3) can be modeled as:

$$O_{trans} = X_{trans} \times m \times d + (X_{switch} \times m + 2 \times X_{port}) \times (d+1) \quad (5)$$

where *d* is the distance from sender to receiver. X_{trans} is the latency of a message transmitting on a link between two switches per byte, X_{switch} is the latency of transmitting through a switch per byte and X_{port} is the latency of passing through switch port. In equation (5), the overhead of transmission is consisted of two parts, one is the latency of transmitting between switches called link cost and another is the latency of transmitting through switches called switch cost.

The *LABS* technique schedules communications of a broadcast in two phases. The first phase transforms original network into several different location based spanning trees. Based on the adjusted spanning trees, the *LABS* constructs optimized binomial-like tree for scheduling contention-free communications to accomplish a broadcast on *HNOW* in the second phase. In the following sections, we describe these two parts in section 4 and section 5, respectively.

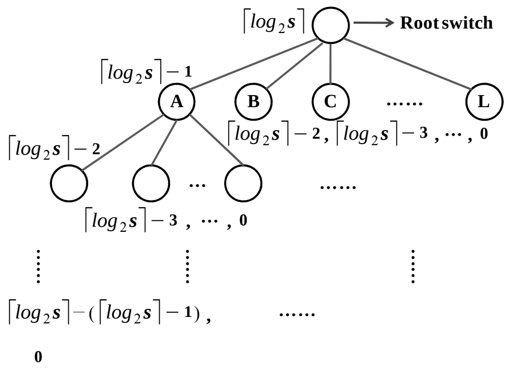
4. Location Oriented Spanning Tree (*LST*) Construction

As traditional methods construct a single spanning tree by using *BFS* or *DFS* for scheduling broadcast communication, it can make the workstations that connected to the switches with large degree have less communication latency to receive the messages. Nevertheless, the workstations connected to the switches that with small degree may waste more communication latency to receive the messages. To resolve this problem, we design the location oriented spanning tree (*LST*) to solve it.

The basic idea of constructing the *LST* is that every switch chooses its neighbor switches as its child switches, and the number of child switches is restricted. The *LST* keeps restricted width in every level of the spanning tree, and we prove that *LST* has restricted depth of the spanning tree in Lemma 1.

Figure 2 gives an example of depth-width restricted *LST*. The number of nodes in the second level is $\lceil log_2 s \rceil$, with **A** is the leftmost child of the root; Following the second level, the number of child switches of node A is $\lceil log_2 s \rceil - 1$, and so on for subsequent levels.

To construct an *LST*, there are some criteria for selecting child switches. Except the rightmost node in every level, higher degree switches are prior to be allocated as child nodes in upper levels. In case two or more switches have the same external degree[1], the internal degree[2] will be adopted as the second criterion. In final case, switches' id will be the last criterion if the switches are with the same conditions for both external and internal degrees.



$\lceil log_2 s \rceil$
Root switch
$\lceil log_2 s \rceil - 1$
A
B
C
......
L
$\lceil log_2 s \rceil - 2$, $\lceil log_2 s \rceil - 3$, ⋯ , 0
$\lceil log_2 s \rceil - 2$
...
......
$\lceil log_2 s \rceil - 3$, ⋯ , 0
$\lceil log_2 s \rceil - (\lceil log_2 s \rceil - 1)$,
......
0

Figure 2. An example of depth-width restricted *LST*.

[1] External degree of a switch is denoted as the number of neighboring switches
[2] Internal degree of a switch is denoted as the number of workstation connected to itself

Lemma 1 *The maximal depth of LST is* $\lceil log_2 \boldsymbol{s} \rceil + 1$, *where* ***s*** *is the number of switches.*

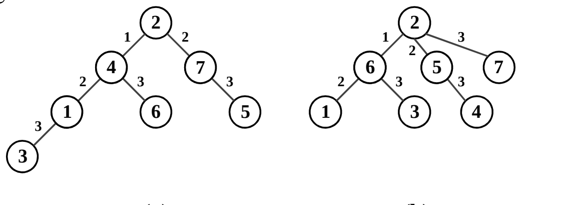
Proof: Due to page limitation, we omit the proof of this lemma in this paper.

5. Scheduling Contention-Free Broadcast

5.1 Binomial-Like Scheduling Tree

Before proceed to illustrate the construction of scheduling tree for broadcast on *HNOW*, we first introduce the binomial-like scheduling tree that will be used in our broadcast scheduling.

Similar to binomial tree construction, the binomial-like scheme partitions an ordered list at the $\lceil (\boldsymbol{n}+1)/2 \rceil$th element recursively to construct the scheduling tree. We use an example to demonstrate the construction of binomial-like representation. Given an ordered list $\{w_2, w_7, w_5, w_4, w_6, w_1, w_3\}$ representing a set of workstations with node w_2 is the sending source. In binomial scheme, the source workstation w_2 selects w_4 as its immediate successor to send the message. Therefore, the list is partitioned into $\{w_2, w_7, w_5\}$ and $\{w_4, w_6, w_1, w_3\}$. On the other hand, in binomial-like scheme, the source workstation w_2 selects w_6 as its immediate successor to send the message. Therefore, the list is partitioned into $\{w_2, w_7, w_5, w_4\}$ and $\{w_6, w_1, w_3\}$. The complete binomial tree and binomial-like tree of this example are shown in Figure 3.



(a) (b)

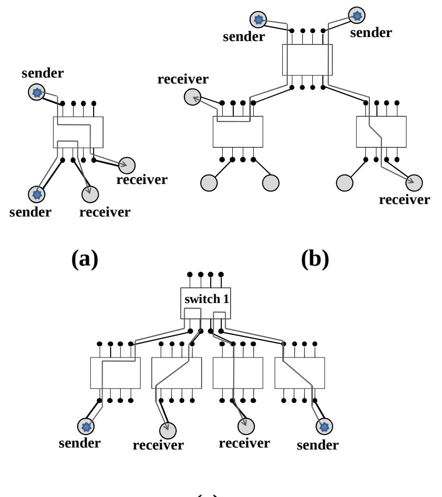
Figure 3. Example of binomial tree and binomial-like tree with seven ordered workstations $\{w_2, w_7, w_5, w_4, w_6, w_1, w_3\}$ (a) the binomial tree (b) the binomial-like scheduling tree, and the number on the link represents communication step

Lemma 2 *Given a numerical list L=* ($w_1, w_2, w_3, \dots, w_m$)*, representing* ***M*** *workstations,* where w_1 *is the sender and* $w_2, w_3, \dots, w_m$ *are receivers. If the number of the receivers is* ***n****, and* $\boldsymbol{n} = \boldsymbol{M} - 1$. *The binomial-like scheduling tree takes* $\lceil log_2(\boldsymbol{n}+1) \rceil$ *communication steps for broadcasting.*

Proof: Due to page limitation, we omit the proof of this lemma in this paper.

5.2 Broadcast Scheduling

Figure 4 demonstrates three circumstances that are link contention-free. Figure 4(a) shows all senders and receivers connect to the same switch, is the first type; in the second type, all senders connect to the same switch, and receivers connect to other switches, as shown in Figure 4(b), the senders can send the messages to their receivers simultaneously without link contention; the third type, there are two messages from different senders' switches to different receivers' switches, as shown in Figure 4(c).



(c)

Figure 4. Three conditions of link contention-free

In the first step of constructing the scheduling tree, it acquires the postorder list of workstations from the *LST*. The postorder list of workstations from the *LST* is an ordered chain, and we use this technique to construct the scheduling tree with link contention-free. In the second step, we choose one workstation with the fastest speed type in every switch as the group head. The members of every group are workstations connected to the same switch. Every group has its sub-tree group size meaning the number of workstations in its sub-tree. In the third step, we adjust these sub-tree groups according to their group sizes decreasingly from left to right. After adjusting the postorder list of workstations, it constructs the first level of the scheduling tree with all group heads. Then every group constructs its scheduling tree called the second level scheduling tree by the binomial-like scheduling until the complete scheduling tree is constructed. In the last step, we rearrange all workstations with faster speed types to be the parent of slower workstations; these workstations are responsible to send the messages to slower others. This movement makes the scheduling tree broadcasting more efficiently.

6. Performance Evaluation

To evaluate the performance of proposed techniques, several parameters are conducted in our experiments, including the number of workstations, network density, message size and network heterogeneity.

As described previously, an *HNOW* consists of network switches and computational workstations. In our experiments, network topology is generated randomly based on given parameters. The unit of the message is *flits*. A large message for broadcasting is 10240 *flits* and a small one is 2048 *flits*. In Table 1, Type 1 represents the fastest speed workstation and Type 8 represents the slowest workstation. In our experiments, fast Ethernet switch is assumed with a 100 *Mbps* (Mega-bits per second) bandwidth and 8 *μsec/port*. The *LABS* and *TWO-VBBS* are conducted in all performance comparisons to verify effectiveness of the proposed techniques.

Table 1. Parameters of Sender and Receiver

Type	S_c (*μsec*)	S_m	R_c (*μsec*)	R_m
1	60.0	0.05	110.0	0.03
2	90.0	0.40	140.0	0.06
3	120.0	0.80	170.0	0.48
4	150.0	1.60	200.0	0.96
5	180.0	2.40	230.0	1.92
6	300.0	3.20	360.0	2.20
7	400.0	3.80	500.0	2.80
8	500.0	4.20	600.0	3.20

6.1 Effects of Message Size

We first study the performance of broadcasting with different message size. The number of workstations is set to 16, 32, 64, 128 and 256 of an *HNOW*. Different speed types of workstations in Table 1 are used to estimate the performance results. Network density[3] is set to 4. In Figure 5, the cost of *LABS* with two speed types is larger than the cost of *TWO-VBBS* with two speed types, but they are close. For four speed types, the cost of *LABS* is almost equal to the cost of *TWO-VBBS*. *LABS* spends less latency than *TWO-VBBS* while the speed type increases to six and eight. The performance of *LABS* outperforms the *TWO-VBBS* scheme with about 12.14% improvements. In Figure 6, the broadcasting message is large and the costs of *LABS* and *TWO-VBBS* with two speed types are very close. While network heterogeneity becomes high, i.e., more different speed types, the cost of *LABS* is less than the cost of *TWO-VBBS*. The *LABS* outperforms the *TWO-VBBS* with about 18.29% improvements.

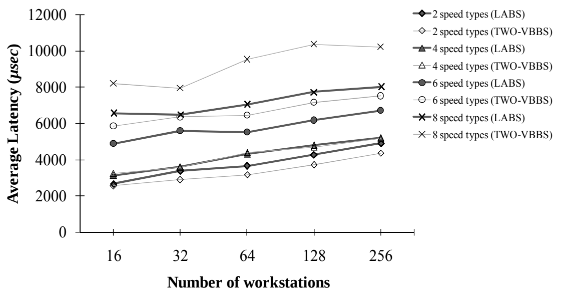


Figure 5. Broadcasting with the small message (2048 *flits*)

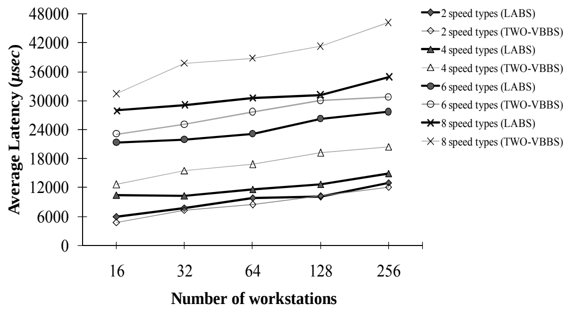


Figure 6. Broadcasting with the large message (10240 *flits*)

6.2 Miscellaneous Comparisons

This section discusses the performance of the two algorithms under different distribution of heterogeneous workstations. The eight different speed types of workstations defined in Table 1 are used again in two tests: In Figure 7, the first one is with the proportion of the 25% slowest workstation and 25% fastest workstation. We use the bell distribution for the eight speed types of workstations in the first test. In the bell distribution, the proportion of workstations from type 1 to type 8 are with 5%, 10%, 15%, 20%, 20%, 15%, 10% and 5%, respectively. The message size is 2048 *flits* and the density is 4. Figure 8 shows that as the number of workstations is increased, the *LABS* has stable performance and is better than the performance of *TWO-VBBS*.

[3] Average number of workstations connected to one switch

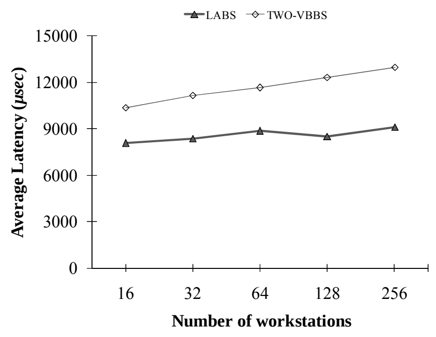


Figure 7. Broadcasting with 25% slowest workstations (Type 8) and 25% fastest workstations (Type 1)

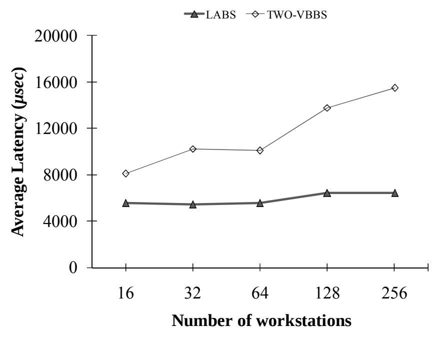


Figure 8. Broadcasting in the bell distribution

7. Conclusions

In this paper, we have presented an efficient broadcast scheme on *HNOW*, termed as Location Aware Broadcast Scheme (*LABS*). Objective of the *LABS* is to design a scheduling tree for broadcasting messages between workstations in the shortest paths and with link contention-free. To broadcast a message over *HNOW*, *LABS* firstly constructs a location oriented spanning tree (*LST*) with restricted depth and width. Based on *LST*, *LABS* designs a scheduling tree according to binomial-like representation, which takes $\lceil log_2(\boldsymbol{n}+1)\rceil$ steps for broadcasting. The scheduling tree is constructed with not only link contention-free but also shortest peer-to-peer communication path. The simulation results show that *LABS* outperforms the *TWO-VBBS* scheduling method in different network configurations. Especially, *LABS* has significant improvements when *HNOW* is with high heterogeneity.

Reference

[1] M. Banikazemi, J. Sampathkumar, S. Prabhu, D. K. Panda and P. Sadayappan, “Communication Modeling of Heterogeneous Networks of Workstations for Performance Characterization of Collective Operations,” *Proceedings of the 8th Heterogeneous Computing Workshop*, pages 125-133, 1999.

[2] M. Gerla, P. Palnati, S. Walton, E. Leonardi and F. Neri, “Multicasting in Myrinet — A High-Speed, Wormhole-Routing Network,” *IEEE Global Telecommunications Conference*, Vol. 2, pages 1064-1068, November 1996.

[3] R. Kesavan, K. Bondalapati and D. K. Panda, “Multicast on Irregular Switch-based Networks with Wormhole Routing,” *Proceedings of the 3th International Symposium on High-Performance Computer Architecture*, pages 48-57, February 1997.

[4] R. Kesavan and D. K. Panda, “Efficient Multicast on Irregular Switch-Based Cut-Through Networks with Up-Down Routing,” *IEEE Transactions on Parallel and Distributed Systems*, Vol. 12, Issue 8, August 2001.

[5] R. Libeskind-Hadas, D. Mazzoni and R. Rajagopalan, “Optimal Contention-Free Unicast-Based Multicasting in Switch-Based Networks of Workstations,” *Proceedings of the Merged 12th International Parallel Processing Symposium and the 9th Symposium on Parallel and Distributed Processing*, pages 358-364, April 1998.

[6] C. Lin, “Heuristic Contention-Free Broadcast in Heterogeneous Networks of Workstations with Multiple Send and Receive Speeds,” *The Journal of Supercomputing*, Vol. 30, Issue 1, pages 37-64, October 2004.

[7] C. Lin and J.-P. Sheu, “Efficient Broadcast in Heterogeneous Networks of Workstations Using Two Sub-Networks,” *International Journal of Parallel Programming*, Vol. 33, Issue 4, August 2005.

[8] C. Lin, Y.-C. Tseng and J.-P. Sheu, “Efficient Single-node Broadcast in Switched-based Network of Workstations with Network Partitioning,” *Proceedings of the 10th International Conference on Computer Communications and Networks*, pages 68-74, 2001.

[9] P. K. McKinley, Y.-J. Tsai and D. F. Robinson, “Collective Communication in Wormhole-Routed Massively Parallel Computers,” *IEEE Computers*, pages 39-50, December 1995.

[10] L. M. Ni and P. K. McKinley, “A Survey of Wormhole Routing Techniques in Direct Networks,” *IEEE Computers*, Vol. 26, Issue 2, pages 62-76, February 1993.

[11] A. Singhal, M. Banikazemi, P. Sadayappan and D. K. Panda, “Efficient Multicast Algorithms for Switch-based Irregular Heterogeneous Networks of Workstations,” *Proceedings of the 15th International Parallel and Distributed Processing Symposium*, April 2001.