On Balancing Traffic Load in Path-Based Multicast Communication

A. Al-Dubai, M. Ould-Khaoua, K. El-Zayyat*, and L. M. Mackenzie

Department of Computing Science
University of Glasgow
Glasgow G12 8QQ, UK.
E-mail: {aldubai, mohamed, lewis}@dcs.gla.ac.uk

**School of Computer Science, Telecommunications and Information Systems*
DePaul University, USA, Email: kelzayyat@cs.depaul.edu

Abstract

Multicast is the most primitive collective capability of any message-passing network. It is itself central to many important parallel applications in Science and Engineering but are also fundamental to the implementation of higher-level communication operations such as gossip, gather, and barrier synchronisation. This paper presents a new efficient multicast path-based algorithm, which can achieve a high degree of parallelism and low communication latency over a wide range of traffic loads in the mesh. To achieve this, the proposed algorithm relies on a new approach that divides the destinations in a way that balances the traffic load on network channels during the propagation of the multicast message. Results from extensive simulations under a variety of working conditions confirm that the proposed algorithm exhibits superior performance characteristics over those of some well-known existing algorithms, such as dual-path, multiple-path, and column-path algorithms.

1. Introduction

Optimising the performance of message-passing multicomputers requires matching inter-processor communication algorithms and application characteristics to a suitable underlying interconnection network. The mesh has been one of the most popular interconnection networks for multicomputers due to many desirable properties, such as ease of implementation, recursive structure, and an ability to exploit the communication locality found in many parallel applications to reduce message latency [1, 2, 9, 15, 20, 21, 22].

The switching method determines the way messages visit intermediate nodes. *Wormhole switching* has been widely used in practice due firstly to its low buffering requirements, allowing for efficient router implementation. Secondly, and more importantly, it makes latency almost independent of the message distance in the absence of blocking. In wormhole switching, a message is divided into elementary units called flits, each of a few bytes for transmission and flow control. The *header* flit (containing routing information) governs the route and the remaining data flits follow it in a pipelined fashion. If a channel transmits the header of a message, it must transmit all the remaining flits of the same message before transmitting flits of another message. When the header is blocked the data flits are blocked in-situ.

Multicast, which refers to the delivery of a message disseminated from a given source to a group of destinations, is one of the most important collective communication operations. It is often required in many scientific computations to distribute large data arrays over system nodes in order, for example, to perform various data manipulation operations. It is also required in control operations such as global synchronisation and to signal changes in network conditions, e.g., faults, image processing, matrix multiplication and graphics on parallel computers.

Multicast latency consists of three components, *start-up latency*, *network latency* and *blocking latency* [3, 4, 12, 14, 18]. The start-up latency is the time incurred by the operating system when preparing a message for injection into the network. The network latency consists of channel propagation and router delays, while blocking latency accounts for delays due message contention over network resources, e.g. buffers and channels.

In current generation machines, the start-up latency is the dominating factor in the cost of communication, being typically in the order of microseconds compared to the network latency, which is in the order of nanoseconds [18]. Blocking latency, on the other hand, depends on the routing algorithm and the generated traffic, and consequently can vary widely depending on instantaneous traffic conditions. Due to the dominance of the start-up latency, much research work has been devoted to the

design of multicast routing algorithms that minimise its effects by reducing the number of start-ups required to complete a multicast operation [1, 3, 5, 9, 17]. Existing path-based routing algorithms usually employ two main approaches to deal with the multicast problem. In the first approach, the algorithms try to reduce the number of start-ups due to its high dominating effects in the overall multicast latency [15, 18]. These algorithms require messages to use long paths to reach their destinations, and as a result are inefficient under high traffic loads.

Algorithms in the second approach tend to use shorter paths, but messages can suffer from high latencies due to the excessive number of start-ups [5, 19]. Moreover, algorithms in both approaches implement a multicast operation in a highly sequential manner and do not consider the variance of message arrival times among the destinations. Therefor, our objective in this paper is the design of efficient multicast path-based routing algorithm that can cover these limitations.

Our proposed algorithm relies on a new grouping approach that balances start-up latency with different network loads so as to achieve high parallelism and low communication latency over a range of network loads. This is of importance particularly for massive communication cases, such as multi-node multicast. The remainder of this paper is organised as follows. Section 2 reviews the existing multicast algorithms that have been proposed in the literature for mesh networks. Section 3 introduces the proposed multicast algorithm. Section 4 compares the performance of the proposed algorithm to some well-known existing multicast algorithms and Section 5 concludes this study.

2. Related work

A path-based routing algorithm known as the Hamiltonian path-based algorithm has been proposed in [15]. In this algorithm, multicast messages are transmitted on one of the two subnetworks, *Hu* and *Hl*, each defining a Hamiltonian path in the network. Fig. 1a illustrates an example of the portions of *Hu* and *Hl* subnetworks that could be by a source node to send its multicast message. The destinations are placed into two groups. One group contains all the destinations that could be reached using the *Hu* subnetwork, and the other contains the remaining destinations, that could be reached using the *Hl* subnetwok. For shorter paths, vertical channels that are not part of the Hamiltonian paths could be used. The short cuts reduce path length as shown in the example of Fig. 1a.

The dual-path algorithm proposed in [15] uses at most two copies of the multicast message. This may decrease the path length for some multicast messages. The multipath algorithm of [15] attempts to reduce path lengths by using up to four copies (or 2*n* for the *n*-dimensional mesh) of the multi-cast message. As per the multi-path multicast algorithm, all the destinations of the multicast message are grouped into four disjoint subsets such that all the destinations in a subset are in one of the four quadrants when source is viewed as the origin. Copies of the message are routed using dual-path routing (see [15] for a complete description).

The dual-path and multi-path schemes provide deadlock free routing and could be used to route unicast and multicast messages simultaneously within a common framework. A major emphasis in the above algorithms has been on reducing the number of start-up latencies that a multicast message can experience. They do not however perform well in the presence of high traffic loads as they generate long paths to complete a multicast operation. To overcome such limitations, the authors in [5] have introduced algorithm, called the column path algorithm [20], which partitions the destinations into sets that can be reached on valid dimension ordered paths.

The column-path algorithm partitions the set of destinations into at most 2*k* subsets (e.g. *k* is the number of columns in the mesh), such that there are at most two messages directed to each column. If a column has one or more destinations in the same row or in rows above that of the source, then one copy of the message is sent to serve all those destinations. Similarly, if a column has one or more destinations in the rows below that of the source, then one copy of the message is sent to serve all those destinations. One copy of the message is sent to a column if all destinations in that column are either below or above the source node; otherwise, two messages are sent to that column [19]. The column-path multicast algorithm uses short paths and performs better in high-loads circumstances. However, it suffers from high network latencies due to its excessive number of start-ups. Furthermore, most existing algorithms implement a multicast operation in a highly sequential manner and do not consider the variance of the messages arrival times at the destinations.

To overcome the limitations of existing solutions, this paper presents a new efficient multicast path-based algorithm, referred to here as the Qualified Groups (*QG* for short). The *QG* can maintain a low start-up, network and blocking latencies while achieving a high degree of parallelism and low communication latency over a wide range of traffic loads in a wormhole-switched network. For the sake of the present discussion, we restrict our attention to the 2D mesh, although all of the results described here could be easily extended to higher dimensions.

The proposed *QG* algorithm can be employed along with any underlying deadlock free routing. In this study, the *e*-cube routing is used as a base routing, as it has been shown in [6, 8] that it is deadlock free and highly capable of exploiting the partitionability of the 2D mesh. Tthe *QG* algorithm is based on a new partitioning approach that can

balance the load on network channels to minimise the delay caused by traffic congestion. Extensive simulations under a variety of conditions will demonstrate the effectiveness of the *QG* algorithm over the well-known dual-path, multiple path, and column-path algorithms, especially in the presence of multiple multicast operations in the network.

3. The proposed multicast algorithm

The multicast problem can be considered formally as follows. A *multicast set* is a couple $(p, Đ)$ where $p \in V$ and $Đ = \{p_1, p_2 ..., p_k\}$, $p_i \in V, i = 1,...,k$. The node p is the source of the multicast message, and the k nodes in $Đ$ are the destinations. Node p has to disseminate the same message to all the destinations in $Đ$. A common problem associated with most existing multicast algorithms is that they can overload the selected multicast path and hence cause traffic congestion. To avoid this problem, the proposed algorithm takes advantage of the partitionable structure of the mesh to distribute the traffic load over several comparable partitions/sub-meshes. These sub-meshes, in turn, implement the multicast independently in a parallel fashion, and balance the traffic load in the network to avoid the congestion problem, and thus considerably reduce the overall communication latency. The proposed algorithm is composed of four phases described as follows.

***Phase* 1:** In this phase, a multicast area is defined as the smallest *n*-dimensional array that includes the source of the multicast message and the set of destinations. The purpose of defining this area is to confine a boundary of network resources that can be utilised during the multicast operation.

***Definition* 1:** *In an n-dimensional mesh with a multicast set* $(p, Đ)$, *a multicast area* G_{MA} *includes the destination nodes* $Đ[(d_1, d_2, ...d_n)]$ *and the source node* $p[d_1, d_2, ...d_n]$, $\forall\ d_i \in \{d_1, d_2, ..., d_n\}$, *has two corners, upper corner* $u_{d_i} = \max(Đ[d_i], p[d_i])$ *and lower-corner*
$l_{d_i} = \min(Đ[d_i], p[d_i]) : mid_{di} = (l_{d_i} + u_{d_i})/2$.

***Phase 2*:** The multicast area G_{MA} is then divided into sub-meshes/groups of comparable sizes. This leads to enables the destination nodes to receive the multicast message in comparable arrival times; i.e., the variance of the arrival times among the destination nodes is minimised.

***Definition* 2:** *In n-dimensional mesh with a multicast set* $(p, Đ)$, *a divisor dimension* Div_{d_i} *for* $Đ$ *satisfies the following condition*
$Div_{d_i} = \min(N_{d_1}, N_{d_2}, ..., N_{d_n})$,

$$N_{di} = \left| Đ[d_i^{\Uparrow}] - Đ[d_i^{\Downarrow}] \right| : Đ[d_i^{\Uparrow}] = \sum_{mid_{d_i}}^{u_{d_i}} Đ[d_i];$$

$$Đ[d_i^{\Downarrow}] = \sum_{l_{d_i}}^{mid_{d_i}-1} Đ[d_i] \qquad (1)$$

The divisor dimension is then used as a major axis for the partitioning scheme in this phase, thus making the partitions comparable in the next phase. After that, the multicast area G_{MA} is divided into a number of disjoint groups/sub-meshes as formulated in the following definition.

***Definition* 3:** *Given a target system* $G = (V, E)$, *and a multicast area* $G_{MA} \subseteq G$, $\forall G_i, G_j : G_i \subseteq G_{MA}$ *and* $G_j \subseteq G_{MA} : G_i \cap G_j = \Phi$.

According to Definition 3, G_{MA} is divided into a number of primary groups/sub-meshes; the primary groups PTN_{pr} refers to the number of the group obtained after dividing the destination nodes over the division dimension, such that

$$PTN_{pr} = \begin{cases} p_t & \exists G_i \subseteq G : G_i = \Phi \\ 2^n & \text{otherwise} \end{cases} \qquad (2)$$

where p_t is an integer, $1 \le p_t < 2^n$

***Phase* 3:** This phase is responsible for qualifying the sub-meshes already obtained by the previous phase for a final grouping. Having primary sub-meshes, PTN_{pr}, for each sub-mesh partition, $G_i \in G_{MA}$, we recursively find the multicast area as defined in Definition 4 and determine the internal distance $Int(G_i)$ for each partition G_i.

$$Int(G_i) = Dist(G_i(p_f), G_i(p_n)) + N_{G_i} \qquad (3)$$

The first component represents the distance between the furthest p_f and nearest node p_n from/to the source node p, respectively, and the second one N_{Gi} is the number of destination nodes that belong to the relevant partition $G_i \in G_{MA}$. We then determine the external distance $Ext(G_i)$.

$$Ext(G_i) = Dist(Gi(p_n), p) \quad (4)$$

Definition 4: Consider $G_i \subseteq G_{MA}$, where $1 < i \le PTN_{pr}$, the total average of minimum weights, W_{av}, for the multicast area G_{MA}, is given by

$$W_{av} = \frac{\sum_{i=1}^{PTN_{pr}} W_m(G_i)}{PTN_{pr}} \quad (5)$$

where the minimum weight for G_i is calculated by

$$W_m(G_i) = Ext(G_i) + Int(G_i) \quad (6)$$

The total average of minimum weights W_{av} is then used to define a qualification point. The qualification point for each group G_i, $QP(G_i)$ is calculated as follows

$$QP(G_i) = \frac{|W_m(G_i) - W_{av}|}{W_{av}} \quad (7)$$

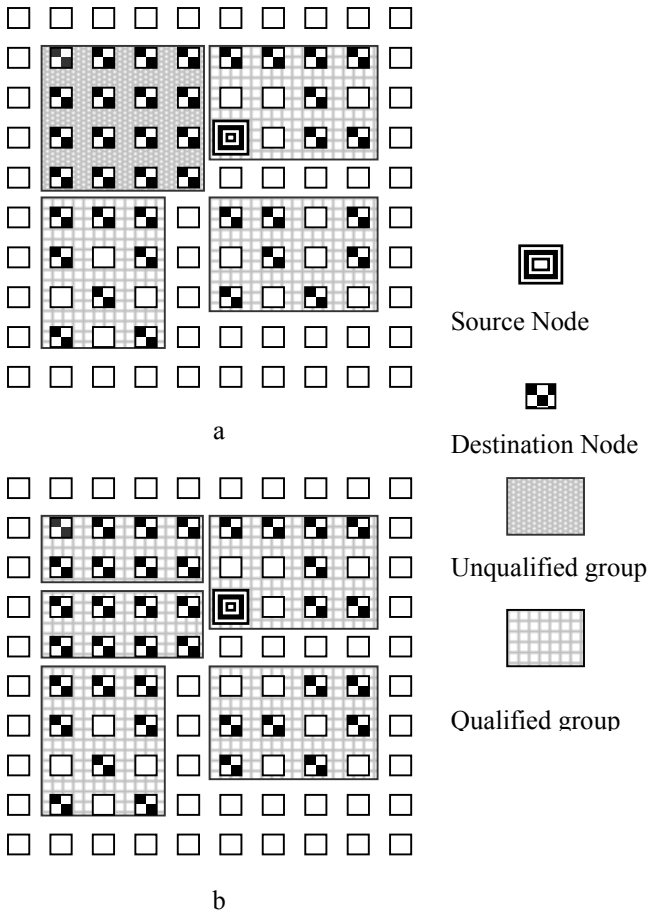
Then, the qualification point for each group is compared to an assumed threshold value T, which is used to set a limit for the partitioning process.

***Definition* 5:** *Consider $G_i \in G_{MA}$, we say that G_i is a qualified sub-mesh/group if and only if its qualification point $(G_i(QP)) \le T$.*

For example, given that the threshold value is $T = 0.5$, each qualified group must hold at last half of the total average weight W_{av} of the groups. Once a group $G_i \subseteq G_{MA}$ does not meet the condition formulated in Definition 8, it is treated as unqualified one. In this case, this group is divided into a number of sub-groups sb, where $2 \le sb \le 2^n$. For instance, for any unqualified group $G_i \subseteq G_{MA}$ in the 2D mesh, it can be divided into 4 groups at maximum even if the new obtained groups are still larger than those that meet the qualification point. In fact, the partitioning process is stopped at this stage in order to reduce the number of comparisons during the qualifying phase. This helps to keep the algorithm simple and leads to low preparation times.

***Phase* 4:** For each sub-mesh/group, a node with the lowest communication cost is given highest priority to receive the multicast message from the source node. In other words, the nearest node for each qualified group is elected so that the source node can deliver the multicast message to the nodes that have been selected in this phase, $\{G_1(p_n),...,G_{PTN_{pr}}(p_n)\}$, with only one start-up time. Concurrently, these selected nodes act as source nodes by delivering the message to the rest destination nodes in their own sub-meshes with only one additional start-up. The main objective behind grouping the destination nodes with this scheme is firstly and most importantly to distribute the traffic load so as to avoid the traffic congestion, which contribute significantly to the blocking latency. Secondly to implement a multicast operation in a high degree of parallelism enabling most of the destination nodes to receive the multicast message in overlapped periods of time.

***Theorem* 1:** The proposed *QG* multicast algorithm can be implemented in the 2D mesh with two start-ups (i.e., message-passing steps) from any source node, irrespective of the number of destination nodes involved.



***Proof*:** Let (p, D) be a multicast set in the 2D mesh. Without loss of generality, let the source p and

destination nodes be distributed as shown in Fig. 1a. The Y dimension is selected as a divisor dimension. Having the divisor dimension, the multicast area is divided into four primary groups: $\{G_1,...,G_4\}$. We then compute the weights of the groups. Based on the rules of the QG algorithm, we find that the groups are qualified except one group (the largest one) as shown in Fig. 1a.

In recursive fashion, the multicast area of this unqualified group is divided into two partitions. The new groups are then compared to the qualified groups already obtained. After qualifying all groups as given in Fig. 1b, the source node sends the message to the nearest destinations in the qualified groups. The source node performs this task with a single start-up latency taking advantage of the multiple-port facility of the system by creating two disjoint paths in this step as shown in Fig. 2a. Concurrently, every node in $\{G_1(p_n),...,G_{PTN}(p_n)\}$ is the new source node and in turn sends the received message to the rest of the destinations in its own group performing as shown in Fig. 2b □.

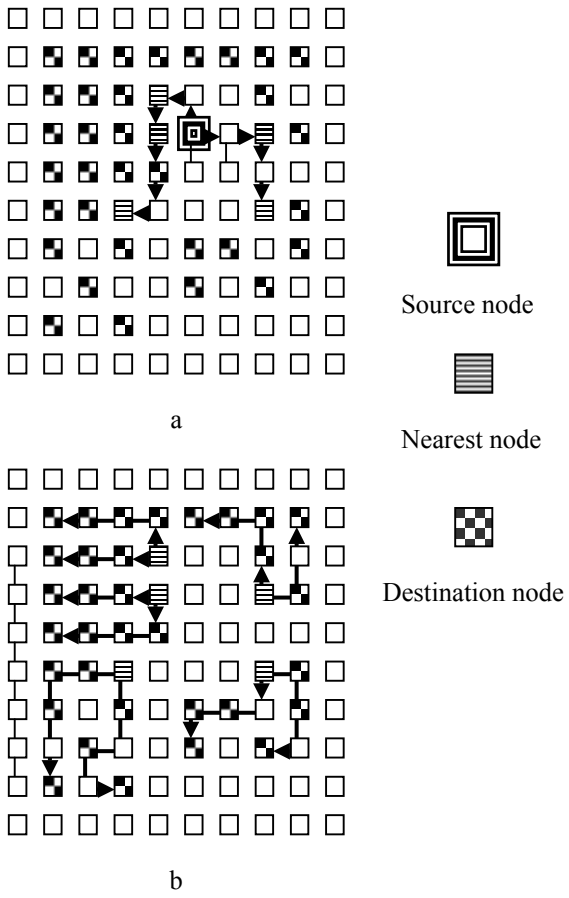


Fig. 2: The first phase (a) and second phase (b) during a multicast operation in the proposed algorithm

4. Performance and comparison

Simulation experiments have been carried out to compare the performance of the proposed *QG* algorithm against well-known multicast algorithms, including dual-path (*DP*), multi-path (*MP*) [4] and column-path (*CP*) [15]. A simulation program was developed to model the multicast operations in the 2D mesh. The program was written in VC++ and built on top the event-driven CSIM 18-package [6]. In the simulator, processes are used to model the active entities of the system, which are executed in a quasi-parallel fashion, and providing a convenient interface for writing modular simulation program. The main program activates a set of CSIM processes, called multicast generators, one for each network node.

Each multicast message is simulated with a pseudo-process that sends messages to the destinations by creating path pseudo-processes. A routing model for each algorithm is used as path processes to determine the channels on which each message should be transmitted. We have used the 2D mesh with four injection channels and four ejection channels. Two unidirectional channels exist between each pair of neighbouring nodes. Each channel has a single queue of messages waiting for transmission. A statistics module gathers information using the mean batch method. In our simulations, the start-up latency has been set at 33 cycles, and the channel transmission time at 1 cycle.

The preparation time (which consists of dividing the destination nodes into appropriate subsets and creating multiple copies of the message as needed, depending on the underling algorithm) of the *DP, MP, CP* and *QG* algorithms are set at 2, 2, 4 and 16 cycles, respectively. A cycle is the time to transmit a flit across a physical channel. The preparation time was deliberately set higher in the *QG* algorithm to reflect the fact that our algorithm requires a longer time to divide the destinations into qualified groups.

In fact, we have found that the conclusions do not change considerably even if this delay factor was set as high as 40 cycles in the *QG* algorithm. The *QG* and *CP* have the most complicated procedures to partition the destination nodes, while the *DP* and *MP* have the easiest procedures. However, unlike the *DP*, *MP* and *CP* algorithms, the preparation time of the proposed algorithm *QG* is distributed over several nodes in the multicast area. In other words, the source node and the selected nodes, $\{G_1(p_n),...,G_{PTN}(p_n)\}$ consume the preparation time and the preparation of the message copies in the selected nodes, $\{G_1(p_n),...,G_{PTN}(p_n)\}$ take place simultaneously to reduce the overhead consumed by the preparation phase.

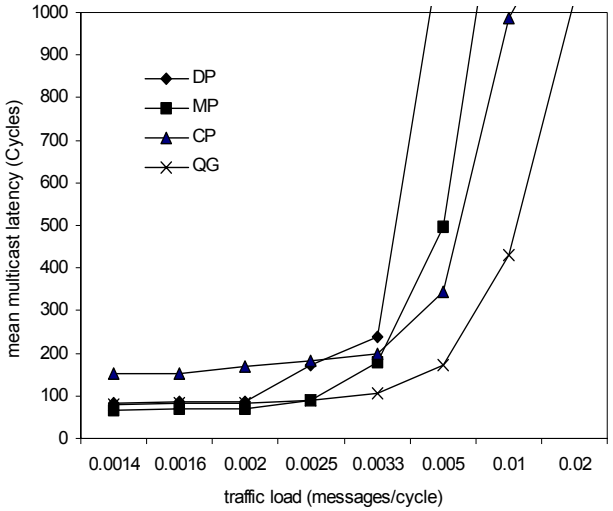


Fig. 3: Performance under different loads in 10×10 mesh with 10 destinations

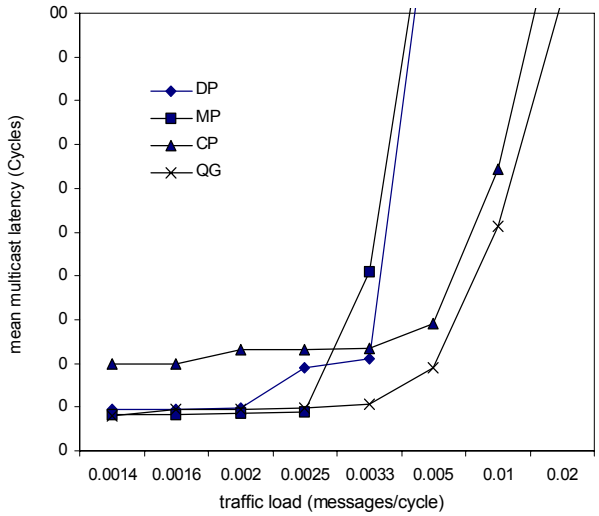


Fig. 4: Performance under different loads in 10×10 mesh with 20 destinations

We have considered a wide range of traffic loads to evaluate the performance of the four multicast algorithms under different working conditions. Two network sizes have been considered N=10×10 and 16×16; we have been limited to such network sizes due to the excessive computing required to run simulation for larger systems. We have fixed the number of average destinations at 10 and the message size at 32 flits. The preparation time of the *DP, MP, CP* and *QG* algorithms have been set at 2, 4, 8 and 16 cycles, respectively.

Figs. 3, 4 and 5 and 6 depict the mean multicast latency against the offered traffic (i.e., traffic generated by the network nodes). At low loads, the start-up latency dominates the communication latency. As a result, the *DP* and *MP* algorithms outperform their *QG* and *CP* counterparts. The *CP* and *QG* algorithms, however, are less sensitive to the increased load than the *DP* and *MP* algorithms. This is due to the fact that the former algorithms introduce less traffic in the network as they use shorter paths; so resources are held for shorter time periods, leading to higher throughput.

Figs. 3 and 5 show that the *MP* algorithm offers a slight improvement over the *DP* algorithm. This is likely because a message in the *MP* reaches the destination nodes with shorter paths. However, in high traffic conditions or larger number of destination nodes (20 destinations) a different scenario could be seen in Figs. 4 and 6. The *QG* algorithm has better performance than its *DP*, *MP* and *CP* counterparts.

Note that, the same conclusion is obtained even when preparation time is set at 40 cycles in the *QG*. The reason is that the proposed algorithm makes the source node responsible for the multicast operation for a shorter time period compared to the other algorithms. Consequently, the multicast period covers two different stages. In the first stage, the source node is responsible only for serving the nearest selected nodes of the destination groups.

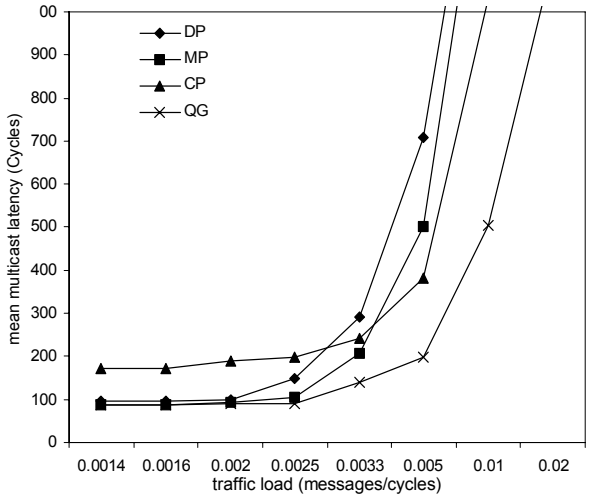


Fig. 5: Performance under different loads in a 16×16 mesh with 10 destinations

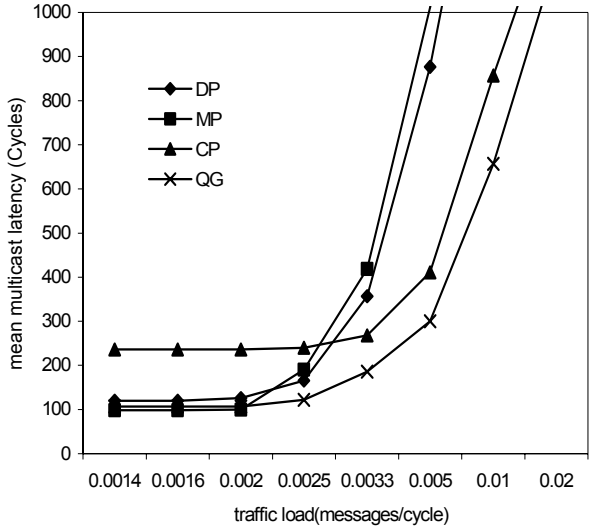


Fig. 6: Performance under different loads in a 16×16 mesh with 20 destinations

The source node then is no longer responsible for the multicast operation as the nearest nodes complete the operation in their destination groups, leading to short multicast periods. However, in the *DP*, *MP* and *CP* algorithms the source node remains engaged until the completion of the multicast operation. This engagement has considerable degrading effect on network performance as a source node uses all its outgoing channels to send out copies of the multicast message.

Until the multicast transmission is complete, flits from other messages that routes through that source node are blocked at that point. The source node becomes a congested point or a hot spot. When the load is high, the source node may throttle system throughput considerably and cause a large increase in the multicast latency.

5. Conclusion and Future Directions

While the existing multicast algorithms are implemented sequentially and are inefficient under high traffic loads, this paper has suggested an efficient multicast algorithm, which overcomes the limitations of the existing algorithms. The proposed algorithm has the main advantage of requiring only two start-ups irrespective of the destination nodes involved. Unlike the previously proposed algorithms, our algorithm achieve a high degree of parallelism and low communication latency over a wide range of traffic loads in the mesh, i.e., most of the network nodes receive the multicast message in parallel. The proposed algorithm relies on a new approach that divides the destinations in a way that balances the traffic load on network channels during the propagation of the multicast message. Moreover, our simulation results have revealed that the proposed algorithm has superior performance characteristics over those of some well-known existing algorithms, such as dual-path, multiple-path, and column-path algorithms.

The next step in our research is to extend our work towards devising new collective communication algorithms such as broadcast and gossip and compare their performance with existing well-known algorithms. Another possible line for future research is to support collective communication in other common multicomputer networks, such as hypercubes and tori.

References

[1] P. McKinley, Y. -J. Tsai, D. Robinson, Collective communication in wormhole-routed massively parallel computers, *IEEE Computer*, vol. 28, no. 12, 39-50, 1995.

[2] J. Duato, S. Yalamanchili, L. Ni, Interconnection networks: An engineering approach, *IEEE Computer Society Press*, 1997.

[3] P. K. McKinley, H. Gu, A. Esfahanian, L. M. Ni, Unicast-based multicast communication in wormhole-routed direct networks, *IEEE TPDS*, vol. 5, no. 12, 1254-1265, 1994.

[4] L. Ni, P. McKinley, A survey of wormhole routing techniques in direct networks, *Computer*, vol. 26, no. 2, pp. 62- 76, Feb. 1993.

[5] D. K. Panda, S. Singal, R. Kesavan, Multidestination message-passing in wormhole *k-ary n*-cube networks with base routing conformed paths, *IEEE TPDS*, vol. 10, no. 1, pp. 76-96, 1999.

[6] H. D. Schwetman, CSIM: A C-based, process-oriented simulation language, Tech. Rep. pp. 80-85, *Microelectronics and Computer Technology Corp*., 1985.

[7] E. Fleury, P. Fraigniaud, Strategies for path-based multicasting in wormhole-routed meshes, *J. Parallel & Distributed Computing*, vol. 60, pp. 26-62, 1998.

[8] Y.-C. Tseng, S.-Y. Wang, C.-W. Ho, Efficient broadcasting in wormhole-routed multicomputers: A network-partitioning approach, *IEEE TPDS*, vol. 10, no. 1, pp. 44-61, 1999.

[9] Y.-J Tsai, P.K. McKinley, An extended dominating node approach to broadcast and global combine in multiport wormhole routed mesh networks, *IEEE TPDS*, vol. 8, no. 1, 41–58, 1997.

[10] P. K. McKinley, C. Trefftz, Efficient broadcast in all-port wormhole-routed hypercubes, *Proc. Int. Conf. Parallel Processing*, pp. 288-291, 1993.

[11] S. Cang, J. Wu, Time-step optimal broadcasting in 3-D meshes with minimal total communication distance, *J. Parallel & Distributed Computing*, vol. 60, pp. 966-997, 2000.

[12] J. Watts, Efficient collective communication on multidimensional meshes with wormhole routing. Tech. Rep. TR-94-19. Dept. Computer Science, Univ. Texas at Austin, June 1994.

[13] D. F. Robinson, P. K. McKinley, C. Cheng, Path based multicast communication in wormhole routed unidirectional torus networks, *JPDC*, vol. 45, 104 - 121, 1997.

[14] D. K. Panda, Issues in designing efficient and practical algorithms for collective communication on wormhole routed systems, *Proc. 1995 Workshop Challenges Parallel Processing*, 8-15, 1995.

[15] X. Lin, L. M. Ni, Deadlock-free multicast wormhole routing multicomputer networks, *Proc. Int. Symp. Computer Architecture*, pp. 116-124, 1991.

[16] M. P. Malumbres, J. Duato, An efficient implementation of tree-based multicast routing for distributed shared-memory multiprocessors, *J. Systems Architecture*, vol. 46, 1019-1032, 2000**.**

[17] Y. Tseng, D. K. Panda, T. Lai, A trip-based multicasting model in wormhole-routed networks with virtual channels, *IEEE TPDS*, vol. 7, no. 2, 138-150, 1996.

[18] P. Mohapatra, V. Varavithya, A hardware multicast routing algorithm for two dimensional meshes, *the Eighth IEEE Symposium on Parallel and Distributed Processing*, pp. 198-205, News Orleans, October 1996.

[19] Rajendra V. Boppana, Surish Chalasani C.S Raghavendra, Resource deadlock and performanceof wormhole multicast routing algorithms, *IEEE TPDS*, vol. 9, no. 6, 535-549, 1998.

[20] S. Wang, Y. Tseng, C. Shiu , J, Sheu, Balancing traffic load for multi-node multicast in a wormhole 2D torus/mesh, the Computer Journal, vol. 44, no. 5, pp. 354-367, 2001.

[21] R. E. Kessler, J. L. Schwazmeier, CRAY T3D: A new dimension for Cray research, *Proc. COMPCON'93,* pp. 176-182, 1993.

[22] D. Lenoski *et al*., The Stanford DASH multiprocessor, *IEEE Computer*, vol. 25, no. 3, 1992.

[23] Z. Liu, J. Duato, Adaptive unicast and multicast in 3D mesh networks, , *Proc. 27th Int. Conf. On System Sciences*, pp. 173-183, 1994.

[24] S. Dobrev, I. Vrto, Optimal broadcasting in tori with dynamic faults, *Parallel Processing Letters*, vol. 12, no. 1, pp. 17-22, 2002.

[25] D. R. Kumar, W. A. Najar, P. K. Srimani, A new adaptive hadrdware tree-based multicast routing in k-ary n-cubes*, IEEE Computer, vol. 50, no. 7, pp. 647-659, 2001.*