Trade-Offs between Latency, Complexity, and Load Balancing with Multicast Algorithms

Ahmed Y. Al-Dubai, *Senior Member, IEEE*, Mohamed Ould-Khaoua, *Member, IEEE*, and Lewis M. Mackenzie, *Member, IEEE*

Abstract—The increasing number of collective communication-based services with a mass interest and the parallel increasing demand for service quality are paving the way toward end-to-end QoS guarantees. Although many multicast algorithms in interconnection networks have been widely reported in the literature, most of them handle the multicast communication within limited performance metrics, i.e., either delay/latency or throughput. In contrast, this study investigates the multicast communication within a group of QoS constrains, namely latency, jitter, throughput, and additional traffic caused. In this paper, we present the Qualified Groups (QGs) as a novel path-based multicast algorithm for interconnection networks. To the best of our knowledge, the QG is the first multicast algorithm that considers the multicast latency at both the network and node levels across different traffic scenarios in interconnection networks. Our analysis shows that the proposed multicast algorithm exhibits superior performance characteristics over other well-known path-based multicast algorithms under different operating conditions. In addition, our results show that the QG can significantly improve the parallelism of the multicast communication.

Index Terms—Interconnection networks, multicast communication, path-based routing, communication algorithms.

1 Introduction

The communication in parallel computers can create bottlenecks for scalable parallel implementations of computationally intensive applications. Collective communication has received considerable attention since it places a high demand on network bandwidth and has a great impact on algorithms execution time. Multicast, in which a source node sends the same message to an arbitrary number of destination nodes in the network, is one of the most useful collective communication operations [1], [2], [3], [6]. Due to its extensive use, efficient multicast is critical to the overall performance of parallel machines [1], [2], [3], [4], [14], [18]. For instance, multicast is frequently used by many important applications such as parallel search and parallel graph algorithms [3], [14]. Furthermore, multicast is fundamental to the implementation of higher level communication operations such as gossip, gather, and barrier synchronization [1], [2], [4]. Ensuring a scalable implementation of a wide variety of parallel applications necessitates efficient implementation of multicast communication. In general, the literature outlines three main approaches to deal with the multicast problem: unicast-based [1], [3], tree-based [3], [17], and path-based [2], [3], [8], [14], [15], [18]. A number of studies have shown that path-based algorithms exhibit superior performance characteristics over their unicast and tree-based counterparts [2], [14], [15], [18].

In path-based multicast, when the flits of a message reach one of the destination nodes in the multicast group, they are copied to local memory while they continue to flow through the node to reach the other destinations [2], [3], [8], [18]. The message is removed from the network when it reaches the last destination in the multicast group.

Although many orthogonal networks have been studied [3], and indeed deployed in practice, none has proved clearly superior in all roles, since the communication requirements of different applications vary widely. Nevertheless, n-dimensional meshes have undoubtedly been the most popular interconnection network used in practice [2], [3], [5], [6], [9], [10], [11], [20], [21] due to their desirable topological properties including ease of implementation, modularity, low diameter, and ability to exploit locality exhibited by many parallel applications [3]. Meshes are suited to a variety of applications including matrix computation, image processing, and problems whose task graphs can be embedded naturally into the topology [3], [6], [10], [18], [19]. Meshes have been used in a number of real parallel machines including the Intel Paragon, MIT J-machine, Cray T3D, T3E, Caltech Mosaic, Intel Touchstone Delta, and Stanford DASH [3]. Recently, among commercial multicomputers and research prototypes, Alpha 21364's multiple processors network and IBM Blue Gene use a 3D mesh. In addition, a mesh has been recently the topology of choice for many high-performance parallel systems and local area networks such as Myrinet-based LANs. More recently, the mesh topology has widely been adopted in network-on-chip technologies, including NOSTRUM, SOCBUS, and RAW (MIT), which are regular mesh architectures [17].

- *A.Y. Al-Dubai is with the School of Computing, Edinburgh Napier University, 10 Colinton Road, Edinburgh, EH10 5DT, UK. E-mail: a.al-dubai@napier.ac.uk.*
- *M. Ould-Khaoua is with the Department of Electrical and Computer Engineering, College of Engineering, Sultan Qaboos University, PO Box 33, Al-Khodh, P.C. 123, Muscat, Sultanate of Oman. E-mail: mok@squ.edu.om.*
- *L.M. Mackenzie is with the Department of Computing Science, University of Glasgow, Glasgow, G12 8QQ, UK. E-mail: lewis@dcs.gla.ac.uk.*

Manuscript received 14 Apr. 2008; revised 2 Dec. 2008; accepted 28 May 2009; published online 22 July 2009.
Recommended for acceptance by F. Lombardi.
For information on obtaining reprints of this article, please send e-mail to: tc@computer.org, and reference IEEECS Log Number TC-2008-04-0160.
Digital Object Identifier no. 10.1109/TC.2009.104.

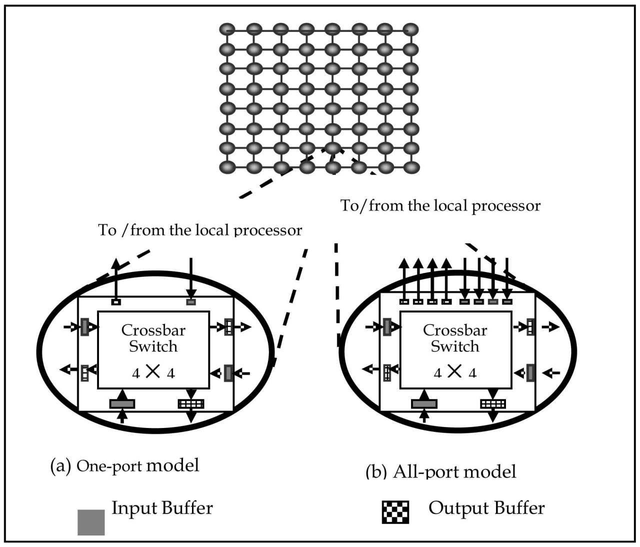


Fig. 1. One-port and all-port node structures in an 8×8 2D mesh.

Unfortunately, most path-based multicast algorithms suffer from several drawbacks, e.g., poor scalability to large network sizes; highly sequential message propagation, caused by either the large number of message passing steps required or long paths used; increased traffic load inside the network; and high variation in the message reception times among the destination nodes. This study presents the Qualified Groups (QGs) as a new path-based multicast algorithm in an attempt to overcome the limitations of the existing solutions. The QG algorithm exhibits a number of desirable properties such as a high degree of scalability and parallelism while imposing low traffic load on network channels, and thus, achieving low multicast latency under a wide range of operating conditions (as will be seen in Section 4). To do so, the QG algorithm relies on a new grouping approach that considers both the effect of start-up latency and that of network load, with the aim of achieving high parallelism and low multicast latency. The rest of the paper is organized as follows: Section 2 presents preliminaries and background on path-based multicast algorithms. Section 3 outlines the proposed multicast algorithm and Section 4 conducts extensive analysis and simulation experiments and Section 5 summarizes this work.

2 Preliminaries and Related Work

Definition 1. *Consider a mesh(V, E), with node set V and edge set E, a multicast set is a couple$(p,Đ)$, where $p \in V$, $Đ = \{p_{1,}p_2, \dots, p_k\}$, and $p_i \in V$, $i = 1, \dots, k$. The node p is the source of the multicast message and the k nodes in Đ are the destinations. To perform a multicast operation, node p disseminates copies of the same message to all the destinations in Đ.*

In the literature, e.g., [3], [4], [18], the mesh is also sometimes referred to as a k-ary n-cube, where k is referred to as the radix and n as the dimension, i.e., has $N = k^n$ nodes, arranged in n dimensions, with k nodes per dimension with bidirectional links. Fig. 1 illustrates a two-dimensional mesh (2D mesh for short) with 8×8 nodes, and Figs. 1a and 1b depict the structure of a node situated in the middle of the network. The discussion can easily be extended to higher dimensional meshes and nodes situated at the corners and edges of the network. A node consists of a processing element (PE) and router. The PE contains a processor and some local memory. A router in the 2D mesh has four inputs and four output channels to connect to its neighboring nodes; two per dimension, one for each direction.

Each router is connected to its local processor by internal channels or ports. When each node has one pair of internal channels, as shown in Fig. 1a, it is referred to as one-port architecture. In this model, one internal channel is used by the processor to inject messages to the network while the other is used to eject messages from the network. A crossbar switch is used to establish a connection between any of the four input channels and any of the four output channels. In this model, when messages destined for the local node arrive at a router on input channels, they are transmitted to the local node sequentially. Fig. 1b shows an *all-port* architecture model, where a node can deal (send/receive) with four messages (equals the number of ports) simultaneously.

Existing path-based algorithms in mesh networks [3], [8], [10], [12], [14], [15] have usually employed two main approaches to deal with the multicast problem. In the first approach, the algorithms try to reduce the number of message passing steps/start-up times due to their high dominating effect in the overall multicast latency. These algorithms are based on dividing the destination nodes into a small number of large groups so that the multicast operation can be handled with few start-up times. For instance, the dual-path and multipath algorithms [3], [10] divide the destinations into two and four disjoint groups, respectively, in the 2D mesh. These algorithms require messages to use long paths to cover the groups serially, resulting in high multicast latency. Due to the generated long paths, the algorithms tend to be inefficient under high traffic loads, as revealed by [2], [3], [15].

In the second approach, the algorithms divide the destination nodes into small groups and often use shorter paths to cover the groups [14], [15]. Despite the fact that the algorithms in this approach perform better under high traffic load compared to those based on the first approach, e.g., [10] messages can suffer from high latencies due to the excessive number of start-ups involved. For most algorithms belonging to this approach, the adopted grouping scheme works for a specific base routing only. For example, the grouping scheme used in the multicast algorithm of [19] works only for the Turn-model-based routing while those used in [10], [11], [15] work only for dimension-ordered routing. To the best of our knowledge, there has not been any study that has proposed a grouping scheme that could work with any base underlying routing algorithm. In general, most existing algorithms incur high multicast latency. This is due to the use of long paths required to cover the groups serially like algorithms under the umbrella of the first multicast approach or those fallen into the second category in which an excessive number of start-ups is involved.

A common problem associated with most existing multicast algorithms is that they can overload the selected multicast path, and hence, cause traffic congestion. This is mainly because most existing grouping schemes [8], [10], [15], [19] do not consider the issue of load balancing during a multicast operation. More importantly, existing multicast algorithms have been designed with a consideration paid only to the multicast latency at the network level, resulting in an erratic variation of the message arrival times at the destination nodes. As a consequence, some parallel applications cannot be performed efficiently using these algorithms, especially those applications that are sensitive to variations in the message delivery times at the nodes involved in the multicast operation. Thus, our objective here is to propose a new multicast algorithm that can overcome the limitations of existing algorithms, and thus, leading to improve the performance of multicast communication in mesh networks. In a previous study [2], a new multicast scheme, the QG, has been proposed for symmetric meshes. Such a scheme has been studied under restricted operating conditions such as specific traffic load, fixed network sizes, and a limited number of destination nodes [2]. In contrast, this paper makes two major contributions. First, the QG is generalized here so that it can deal with symmetric and asymmetric networks. Second, unlike in many previous similar works, this study considers the issue of multicast latency at both the network and node levels across different traffic scenarios, considering several important issues with relation to QoS during the communication phase. Moreover, we added sections related to the complexity and performance under unicast, multicast traffic patterns, explaining more in-depth a number of essential issues and new significant results.

3 The Qualified Groups Algorithm

In an attempt to avoid the problems of existing multicast algorithms, this section presents QG as a new path-based multicast algorithm. The *QG* algorithm takes advantage of the partitionable structure of the mesh to divide the destination nodes into several groups of comparable sizes in order to balance the traffic load among these groups, which leads to avoid the congestion problem in the network. The groups, in turn, implement multicast independently in a parallel fashion, which results in reducing the overall communication latency. In general, the proposed algorithm is composed of four phases, which are described below. For the sake of the present discussion and for illustration in the diagrams, we will assume that messages are routed inside the network according to dimension order routing. We have adopted this routing due to the fact that this form of routing is simple, deadlock, and livelock free, resulting in a faster and more compact router when the algorithm implemented in hardware, [3], [15]. However, the *QG* algorithm can be used along any other underlying routing scheme, including the well-known Turn model and Duato et al.'s adaptive algorithms [3], since the grouping scheme in *QG* can be implemented irrespective of the underlying routing scheme (in the algorithmic level), which is not the case in most existing multicast algorithms in which destination nodes are divided based on the underlying routing used (in the routing level) [8], [10], [15], [18]. It is worth mentioning that such a research line will be investigated further in our future works.

***Phase* 1.** In this phase, a multicast area is defined as the smallest n-dimensional array that includes the source of the multicast message as well as the set of destinations. The purpose of defining this area is to confine a boundary of network resources that need to be employed during the multicast operation. In Definition 2, we describe the boundaries of the multicast area.

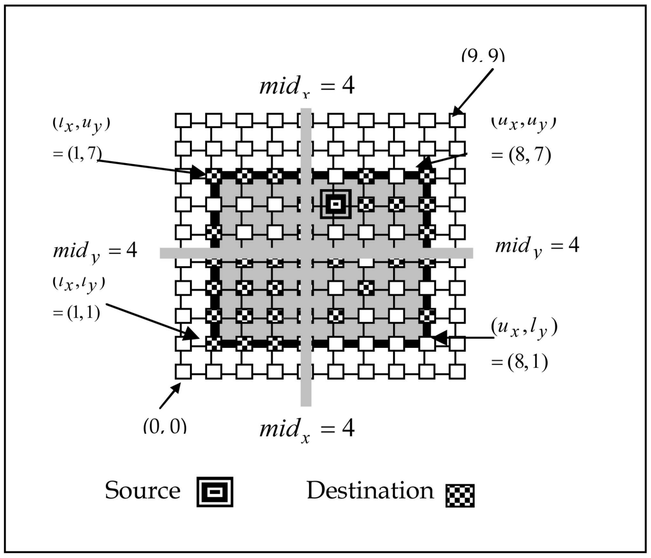


Fig. 2. Example of a multicast area of the QG algorithm.

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

$$mid_{d_i} = \begin{cases} (l_{d_i} + u_{d_i})/2 \text{ if } (l_{d_i} + u_{d_i}) \text{ is even,} \\ ((l_{d_i} + u_{d_i}) - 1)/2 \text{ if } (l_{d_i} + u_{d_i}) \text{ is odd.} \end{cases}$$

***Phase* 2.** The multicast area G_{MA} is then divided into groups. The objective behind grouping the destination nodes is to distribute the traffic load over the multicast area in order to avoid traffic congestion, which contributes significantly to the blocking latency. Besides, grouping enables the destination nodes to receive the multicast message in comparable arrival times, i.e., this helps to keep the variance of the arrival times among the destination nodes to a minimum. For the sake of illustration, let the system be a 10×10 2D mesh, the multicast area is determined as depicted in Fig. 2.

Definition 3. *In an n-dimensional mesh with a multicast set* $(p, Đ)$, *a divisor dimension* Div_{d_i} *for* $Đ$ *satisfies the following condition*

$$Div_{dj} = \min(N_{d_1}, N_{d_2}, \ldots, N_{d_n}),\ N_{d_j} = \left|Đ\left[d_j^{\Uparrow}\right] - Đ\left[d_j^{\Downarrow}\right]\right| :$$

$$Đ[d_j^{\Uparrow}] = \sum_{mid_{dj}+1}^{u_{d_j}} Đ[d_j];\ Đ[d_i^{\Downarrow}] = \sum_{l_{dj}}^{mid_{dj}} Đ[d_j].$$

Note that if $N_{d_1} = N_{d_2}$, d_1 is given a higher priority, i.e., a higher priority is given based on the ascending order of the dimensions. For instance, if $N_x = N_y = N_z$, X dimension will be considered as a divisor dimension. The divisor dimension is used as a major axis for the grouping scheme

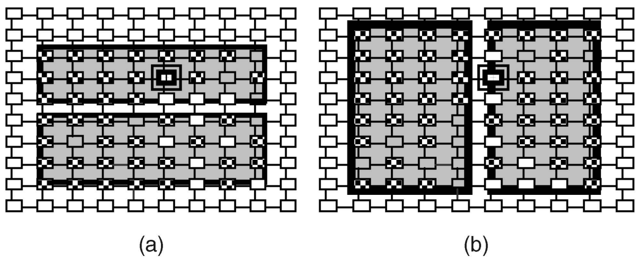


Fig. 3. A divisor dimension determined in the second phase.

in this phase. For instance, let the destination nodes $Đ$ in the 10×10 2D mesh be distributed according to the two scenarios depicted in Figs. 3a and 3b, respectively. According to Definition 3, the Ydimension is the divisor dimension Div_Y for $Đ$ in Fig. 3a, whereas Fig. 3b shows an opposite case, i.e., the X dimension is the divisor dimension Div_X for $Đ$. The algorithm for computing the multicast area and division dimension is shown in Fig. 4. The multicast area G_{MA} is then divided into a number of disjoint groups as formulated in the following definition.

Definition 4. *Given an n-dimensional mesh with a multicast set* $(p, Đ)$ *and a multicast area:*

$$G_{MA}, \forall G_i, G_j : G_i \subseteq G_{MA} \text{ and } G_j \subseteq G_{MA} \rightarrow G_i \cap G_j = \Phi.$$

According to Definition 4, G_{MA} is divided into a number of primary groups as given in (1), where g_{pr} refers to the number of primary groups obtained after dividing the destination nodes over the division dimension, such that

$$g_{pr} = \begin{cases} p_t & \text{if } \exists G_i \subseteq G_{MA} : G_i = \Phi, \\ 2^n & \text{otherwise,} \end{cases} \quad (1)$$

where p_t is an integer that refers to the number of primary groups such that $1 \leq p_t < 2^n$.

***Phase* 3.** This phase is responsible for qualifying the groups already obtained in the preceding phase for a final grouping. Having obtained the primary groups g_{pr}, we recursively find the multicast area for each group, $G_i \subseteq G_{MA}$, as defined in Definition 4, and determine the internal distance $Int(G_i)$ for each group G_i

$$Int(G_i) = Dist(P_f(G_i), p_n(G_i)) + N_{G_i}, \quad (2)$$

where $Dist$ refers to the Manhattan distance in which the distance between two nodes, for instance, $(p1_x, p1_y)$ and $(p2_x, p2_y)$, is given by $Dist(p1, p2) = |(p1_x - p2_x)| + |(p1_y - p2_y)|$. Note that, the first term $Dist(P_f(G_i), p_n(G_i))$ in the above equation represents the distance between the farthest p_f and the nearest node p_n in a group G_i from/to the source node p, respectively, while the second term N_{G_i} represents the number of destination nodes that belong to the relevant group $G_i \subseteq G_{MA}$. We then determine the external distance $Ext(G_i)$

$$Ext(G_i) = Dist(p_n(G_i), p). \quad (3)$$

The minimum weight W_m for a group $G_i, 1 < i \leq g_{pr}$, where g_{pr} refers to the number of primary groups, is then calculated by

$$W_m(G_i) = Ext(G_i) + Int(G_i). \quad (4)$$

```
Procedure: the multicast area $G_{MA}$ and the divisor dimension $Div_{di}$ in the multicast area in n-dimensional meshes.
Input: A multicast set $(p, Đ)$, destination nodes and a multicast message M. Output: the multicast area $G_{MA}$ and the divisor dimension $Div_{d_i}$

Begin
  $\forall$ $d_i \in \{d_1, d_2, ..., d_n\}$ {
    $u_{d_i} = \max(Đ[d_i], p_{d_i})$ , $l_{d_i} = \min(Đ[d_i], p_{d_i})$
    and $mid_{d_i} = \begin{cases} (l_{d_i} + u_{d_i})/2 & \text{if } (l_{d_i} + u_{d_i}) \text{ is even} \\ ((l_{d_i} + u_{d_i}) - 1)/2 & \text{if } (l_{d_i} + u_{d_i}) \text{ is odd} \end{cases}$
    $N_{d_i} = \left| Đ[d_i^{\Uparrow}] - Đ[d_i^{\Downarrow}] \right|$ :
    $Đ[d_i^{\Uparrow}] = \sum_{mid_{d_i}}^{u_{d_i}} Đ[d_i]$ and $Đ[d_i^{\Downarrow}] = \sum_{l_{d_i}}^{mid_{d_i}-1} Đ[d_i]$
  }
  Select the divisor dimension $Div_{d_i} = \min(N_{d_1}, N_{d_2}, ..., N_{d_n})$
  Find the primary groups $= \{G_1, G_2, ...G_{g_{pr}}\}$
End
```

Fig. 4. Computing the multicast area G_{MA} and divisor dimension Div_{d_i} in the QG algorithm.

Definition 5. *Given a multicast area G_{MA} and $G_i \subseteq G_{MA}$, where $1 < i \leq g_{pr}$, the average of the minimum weights W_{av}, for the multicast area G_{MA}, is given by*

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

Definition 6. *Given a multicast area $G_{MA}, G_i \subseteq G_{MA}$, and W_{av}, the qualification point $QP(G_i)$ for each group is calculated as follows:*

$$QP(G_i) = \frac{(W_m(G_i) - W_{av})}{W_{av}}. \quad (6)$$

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

Definition 7. *Given a multicast area G_{MA} and $G_i \subseteq G_{MA}$, we say that G_i is a qualified group if and only if its minimum weight $W_m(G_i) \leq W_{av}$ or if its qualification point $(QP(G_i)) \leq TD$.*

For example, given that the threshold value is $TD = 0.5$, each qualified group must hold at least half of the total average weight W_{av} of the groups. Once a group $G_i \subseteq G_{MA}$ does not satisfy the condition formulated in Definition 7, it is treated as an unqualified group. In this case, this unqualified group is divided into two subgroups based on its division dimension. If the new groups are qualified, the partitioning process is terminated. Otherwise, the unqualified group is divided into a number of subgroups sb, where $2 \leq sb \leq 2^n$. Note that, in the worst scenario, we might obtain a large group of destinations compared to the other obtained groups. Here, the QG will divide the group into subgroups (at maximum 2^n, i.e., four groups for 2D mesh at maximum) with the aim of obtaining more balanced/comparable groups. This is because that we aim at obtaining comparable/balanced groups but not at the expense of the latency or the preparation time.

```
Algorithm: The Qualified Groups multicast
Inputs: A multicast set $(p, Đ)$ , threshold value $TD$ , a multicast
  message $M$
Output: $Đ$ receive the multicast message $M$

Begin
  Step 1: Find the multicast area $G_{MA}$ and primary sub-meshes $g_{pr}$
  $\forall G_i \subseteq G_{MA}$ {
     Find the multicast area;
     $Int(G_i) = Dist(P_f(G_i), p_n(G_i) + N_{G_i})$
     $Ext(G_i) = Dist(p_f(G_i), p)$
     $W_m(G_i) = Ext(G_i) + Int(G_i)$
  }
  Step 2: Qualify the groups and perform multicast
  Find the average weight of the obtained groups $W_{av}$ and the $QP(G_i)$
  $\forall G_i \subseteq G_{MA}$ {
     $sub = 1$;
       While (! Qualify and $sub \le 2^n$ ) //do partition until the
         all groups become qualified{
           If $(QP(G_i)) \le TD$ then {
           $G_i$ is a qualified group
             Find the $p_n(G_i)$ //find the nearest node to the source
           for every qualified group
           }
           Else {
           $sub := sub + 1$;

           $G_i = \frac{G_i}{sub}$; $g := g_{pr} + 1$;
           Find $QP(G_i)$ of the new groups
           }
        $i = i + 1$;
       }
     send out $M$ to $\{p_n(G_1), p_n(G_2), \ldots p_n(G_g)\}$
     $\forall$ $p_n(G_i)$ in parallel do
    send out $M$ to $(N_{G_i} - p_n(G_i))$ // to the rest of the nodes in its
    own group
  }
End
```

Fig. 5. Description of the proposed QG multicast algorithm.

Phase 4. For each group resulting from Phase 3, the nodes that have the lowest communication cost, in terms of distance from the source node, are selected as the representative nodes of the qualified groups that can receive the multicast message from the source node. In other words, the nearest node for each qualified group is elected so that it could be sent the multicast message with a single start-up only. Concurrently, the representative nodes act as "source" nodes by delivering the message to the rest of the destination nodes in their own groups with one additional start-up time only. The main objective behind grouping the destination nodes is first, and most importantly, to balance the traffic load inside the network in order to avoid the congestion problem. Second, the adopted grouping scheme helps to implement a multicast operation with a high degree of parallelism and enables most of the destination nodes to receive the multicast message in overlapped periods of time. For the sake of illustration, let $(p, Đ)$ be a multicast set in a 2D mesh, and the destination nodes be distributed as shown in Fig. 2.

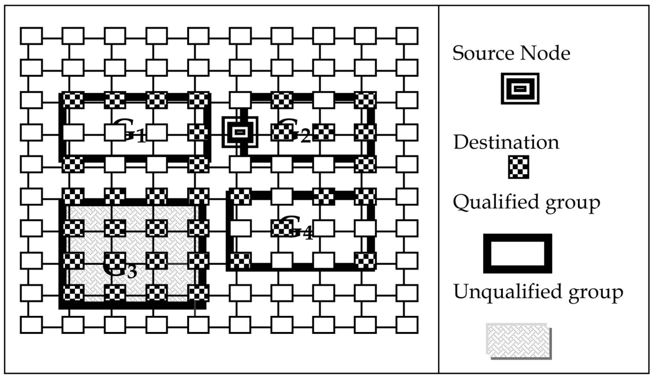


Fig. 6. An example of primary groups in the QG multicast algorithm.

Due to the particular location of the destination nodes in this case, the Y dimension is selected as the divisor dimension $(N_y = 9 < N_x = 11)$, based on Definition 3. Having defined the divisor dimension, the multicast area is divided into four primary groups: $\{G_1, \ldots, G_4\}$ as illustrated in Fig. 6. The weights and the qualification point of the groups are then computed using (2)-(6).

For clarity, the minimum weight of $G_1, W_m(G_1) = Int(G_1) + Ext(G_1) = (5 + 7) + 1 = 13$. Similarly, $W_m(G_2)$, $W_m(G_3)$, and $W_m(G_4) = 11$, 25, and 13, respectively. Therefore, the average of the minimum weights $W_{av} = 15.5$. Based on Definition 7, G_1, G_2, and G_4 are already qualified groups as their minimum weights are less than the average weight. However, the weight of G_3, $W_m(G_3) = 25$ is larger than average weight. Using Definition 6, the qualification point $QP(G_3)$ of G_3 can be calculated as follows:

$$QP(G_3) = \frac{(25 - 15.5)}{15.5} = 0.6.$$

Using a threshold value $TD = 0.5$, we find in this example that G_3 (as shown in Fig. 5) is not qualified (based on Definition 7). Therefore, the multicast area of this unqualified group is divided into two further subgroups based on the division dimension (X in this case). The new subgroups are then compared to the qualified groups already obtained. After qualifying all the groups, as given in Fig. 7, the source node sends the message to the representative nodes in the qualified groups.

The source node performs this operation 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. 8. Concurrently, every representative node in each group acts as a source node, and in turn, sends the message to the rest of the destinations in its own group as illustrated in Fig. 9. In [10] and [8], the authors have shown that each message passing step requires a message preparation phase to sort k addresses with a minimum software cost of O($k \times \log k$). As a consequence, many research works have focused mainly on keeping the time complexity of the preparation phase polynomial. This is of importance especially when a large number of destination nodes K are involved, which may require a longer preparation time than the actual message transmission time [3]. Taking this issue into account, we have adopted QG with polynomial time complexity during the preparation phase.

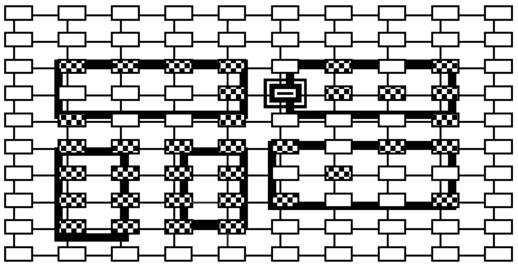


Fig. 7. An example of qualified groups in the multicast area using the QG algorithm.

Below, the time complexity is derived. In contrast, the worst scenario is that some groups are not qualified and need to be partitioned further. According to the rules of the *QG* algorithm, each group could be partitioned m times, where $1 \leq m \leq 2^d$. Thus, the time complexity of the grouping process is $O(m + (k \log k))$, where the first component $O(m)$ represents the partitioning time and the second one $O(k \log k)$ refers to the time required to sort the destination nodes.

Theorem 1. *The preparation time of the QG multicast algorithms in $N \times N$ mesh with k destination nodes is:*

$$O((m + (k \log k)) + (g \log g) + \max\{(k_1 \log k_1), \dots, (k_g \log k_g)\});$$
$$k_i = N_{G_i},$$

g is the number of the groups.

Proof. Let us consider the major time-consuming parts of the grouping scheme in order to derive an estimate of the time complexity of the proposed *QG* algorithm. The grouping scheme consists of three stages. In the first stage, the primary groups are constructed, which could result in dividing the destination nodes into 2^n groups in an n-dimensional mesh. In the best possible case, each group is already qualified. In this case, there is no need for further partitioning process, i.e., like in the dual path and multiple path schemes, the *QG* algorithm requires only $O(k \log k)$ as a preparation time in this stage [10].

The new subgroups are then compared to the qualified groups already obtained. After qualifying all the groups, as given in Fig. 7, the source node sends the message to the representative nodes in the qualified groups. The source node performs this operation with a single start-up latency taking advantage of the multiple-port facility of the system by creating two disjoint paths

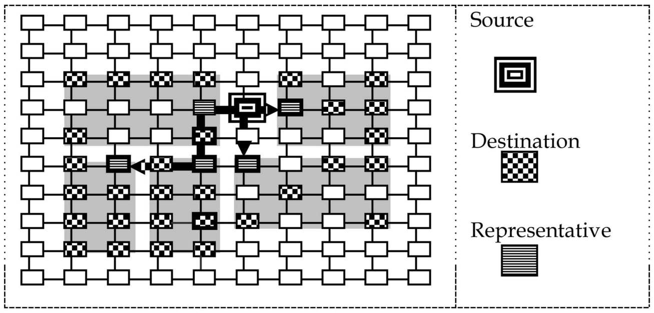


Fig. 8. The first communications step (which occurs in phase 3) in the QG multicast algorithm.

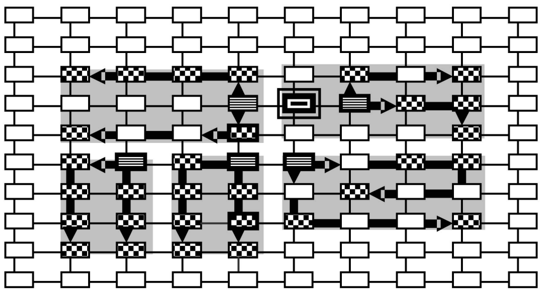


Fig. 9. The second communications step (which occurs in phase 4) in the QG multicast algorithm.

in this step as shown in Fig. 8. Concurrently, every representative node in each group acts as a source node, and in turn, sends the message to the rest of the destinations in its own group as illustrated in Fig. 9. In the second stage, the source node prepares the multicast message to be sent to the representative nodes. In this stage, the time complexity is $O(g \log g)$, where g refers to the number of obtained groups.

Finally, each representative node prepares the message to be disseminated to the rest of the destination nodes. This preparation time can be approximated to be $\max\{O(k_1 \log k_1), \dots, (k_g \log k_g)\}$, where $k_i = N_{G_i}$; N_{G_i} represents the number of destination nodes that belong to the relevant group $G_i \subseteq G_{MA}$. Thus, the total time complexity of the *QG* multicast algorithm in the worst case is $O((m + (k \log k)) + (g \log g) + \max\{(k_1 \log k_1), \dots, (k_g \log k_g)\})$. □

Clearly, this algorithm has polynomial time complexity. Table 1 shows the time complexity of the four algorithms: *DP*, *MP*, *CP*, and *QG* (the time complexity of the *DP*, *MP*, and *CP* has been discussed in [10], [15], respectively).

4 Performance Analysis

Having devised a new multicast algorithm, this section compares the performance of the proposed QG algorithm against the well-known path-based multicast algorithms. To this end, results from simulation experiments are presented to analyze the performance of the *QG* algorithm and compare it against that of dual path (*DP*) [10], multipath (*MP*) [10], and column path (*CP*) algorithms [15]. In fact, the *DP*, *MP*, and *CP* multicast algorithms have been considered in this study due to two main reasons. First, these

TABLE 1
The Preparation Time of the Multicast Algorithms in $N \times N$ Mesh with k Destinations

Algorithm	The preparation time (ordering and grouping destination nodes) of the multicast algorithms
DP	$O(k \log k)$
MP	$O(k \log k)$
CP	$O(k \log k) + \max\{(k_1 \log k_1),...(k_i \log k_i)\}$, $1 \leq k_i \leq N-1$
QG	$O((m + (k \log k)) + (g \log g) + \max\{(k_1 \log k_1),...,(k_g \log k_g)\})$; $k_i = N_{G_i}$, g is the number of the groups

TABLE 2
Notation Used in the Analysis

Notation	Discription
T , T_s	Multicast latency and startup lateny, respectively
M	Number of message passing steps
D	Total distance per step
β	Transmit time (flit/chanel)
c	Network radix
L	Message size
n	Network dimension(s)

algorithms are among the most well-known algorithms and also have shown superiority against a considerable number of multicast algorithms. Second, these algorithms belong to the two main grouping strategies which have been discussed before, i.e., *DP* and *MP* lie under the umbrella of the first grouping strategy, while *CP* is based on the second one as discussed before.

Our comparison study considers the multicast latency at both the network and node levels. The multicast latency at the network level refers to the total time required to complete the multicast operation to reach all the destination nodes, while latency at the node level refers to the time required by a given destination node to receive the multicast message. Most existing multicast algorithms have focused on the multicast latency at the network level only, with a little consideration for the variation in message arrival times at the node level, resulting in an erratic variation of the message arrival times at the destination nodes. To the best of our knowledge, this study is the first to consider the issue of multicast latency at both the network and node levels. The results presented in this section will show that the *QG* algorithms have superior performance characteristics, in terms of multicast latency at both the network and node levels, over those of the existing *DP*, *MP*, and *CP* algorithms.

4.1 Latency in the Absence of Contention

Expressions for the multicast latency in each of the four algorithms, *QG*, *DP*, *MP*, and *CP*, are presented that could be used to analyze the effects of important parameters, such as the message length, network size, and start-up latency, on system performance as given in Table 2.

Definition 8. *In the absence of message contention in the network, the multicast latency T for a message length of L flits can be estimated as*

$$\sum_{i=1}^{M}\left(T_{s(i)}+\sum_{j=1}^{D}\beta_{(i,j)}+L\beta_{(i,j)}\right), \tag{7}$$

where M is the number of message passing steps required to deliver a multicast message to the destination nodes, T_s is the start-up latency, β is the time required to transmit a flit on a channel, and D is the total distance traversed by the message. Under the uniform traffic pattern in an n-dimensional mesh with a network radix c, i.e., the number of nodes per dimension, the average number of channels that a message crosses along each dimension and the network, $\overline{c}$ and D, respectively, are given in [13]

$$\overline{c}=\begin{cases}\frac{c}{4} & c \text{ is even,}\\ \frac{1}{4}(c-\frac{1}{c}) & \text{otherwise,}\end{cases} \tag{8}$$
$$D=n\overline{c}.$$

It is worth mentioning that (7) has been widely adopted in similar studies [8], [10], [15].

4.1.1 Latency in the QG Algorithm

Theorem 2. *In the absence of message contention in the network, the multicast latency T_{QG} in QG can be approximated by*

$$\begin{aligned}T_{QG}=2(T_s+\beta L)+\beta\times(Dist(p,p_f))+\beta\\ \times\max(Dist(p_n(G_1),p_f(G_1)),\ldots,Dist(p_n(G_g),\\ p_f(G_g))).\end{aligned} \tag{9}$$

Proof. As described in the previous section, two message passing steps are required to carry out the multicast operation in the *QG*. According to the rules of the *QG* (discussed in the previous section), in the first message passing step, the message sent from the source node p needs to cover the selected representative nodes from each qualified group, $\{p_n(G_1),\ldots,p_n(G_2),\ldots p_n(G_g)\}$, where $p_n(G_i)$ and g refer to the nearest node to the source in a qualified group G_i and the number of the groups, respectively. To implement this step, the source node p requires T_s cycles as a start-up time to initiate the message. In this message passing step, the message traverses $Dist(p,p_f)$ channels, where p_f refers to the farthest node in the qualified groups from the source node p. The representative nodes that have received the message in the first message passing step act as "new" source nodes and implement the multicast operation in their own qualified groups. A message from each representative node $\{p_n(G_1),\ldots,p_n(G_2),\ldots,p_n(G_g)\}$ traverses $Dist(p_n(G_i),p_f(G_i))$ channels, where $p_f(G_i)$ refers to the farthest node in the group G_i. This message passing step requires T_s cycles to initiate the message. □

4.1.2 Latency in the DP and MP Algorithms

The DP and MP have been discussed in [10]. Given a multicast $(p,Đ)$, the *DP* multicast algorithm proposed by Lin et al. [10] splits $Đ$ into two sets $Đ_u$ and $Đ_l$, where $Đ_u=\{v_1,v_2,\ldots,v_r\}$ includes the destinations whose addresses are below the source node address, v_r refers to the nearest node from p; $Đ_l=\{u_1,u_2,\ldots,u_s\}$ includes the destinations whose addresses are above the source node address, u_s refers to the farthest node from p. Then, it constructs two messages, the first carries the addresses of $Đ_u$ and the second carries those of $Đ_l$. The source node p sends these two messages simultaneously to the destination nodes with communication time T_{DP}, given by [8], [10]:

$$\begin{aligned}T_{DP}=T_s+\beta L+\beta\times\max(Dist(p,v_r)\\ +\sum_{i=r-1}^{1}Dist(v_{i+1},v_i),Dist(p,u_1)+\sum_{i=1}^{s-1}Dist(u_i,u_{i+1})).\end{aligned} \tag{10}$$

From the *DP* algorithm, the authors of [10] have derived the multipath algorithm (*MP*) for the 2D mesh by splitting Đ_l and Đ_u into further two subsets, Đ_l^1, Đ_l^2 and Đ_u^1, Đ_u^2, respectively. The message is then sent to the four sets simultaneously through the output ports of the source node p. The communication latency of the multipath algorithm can be approximated by T_{MP}:

$$T_{MP} = \begin{cases} T_{DP} \text{ if only } \text{Đ}_l \text{ and } \text{Đ}_u \text{ groups are obtained,} \\ \max(T_{G_1}, \ldots, T_{G_4}) \text{ otherwise for 2D mesh,} \\ \max(T_{G_1}, \ldots, T_{G_{2n}}) \text{ otherwise for n-dimensional mesh,} \end{cases} \tag{11}$$

$$T_{G_i} = \begin{cases} T_s + \beta L + \beta \times \sum_{i=r-1}^{1} Dist(p, v_i) \text{ if } G_i \subseteq \text{Đ}_l^1 / G_i \subseteq \text{Đ}_l^2, \\ T_s + \beta L + \beta \times \sum_{i=1}^{s-1} Dist(p, u_i) \text{ if } G_i \subseteq \text{Đ}_u^1 / G_i \subseteq \text{Đ}_u^2. \end{cases} \tag{12}$$

In the *MP* algorithm, latency can be the same of the *DP* algorithm if the destinations belong to the same subset, but it can also be much smaller depending on the partition of the destinations as formulated in (9)-(11).

4.1.3 Latency in the CP Algorithm

The CP algorithm partitions the set of destinations into at most $2k$ subsets in a $(k \times k)$-node mesh (k is the number of columns in the mesh), such that there are at most two copies of the multicast message sent to each column [15]. If a column contains 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 [15]. The multicast latency T_{CP} consumed by the CP is given by

$$T_{CP} = \text{ƛ}(T_s + \beta L + \beta \times \max(Dist(p, p_{c_1}), \ldots, Dist(p, p_{c_\hbar})), \tag{13}$$

where $1 \leq \text{ƛ} \leq \frac{2k}{\psi}$, ψ is the number of ports, p_{c_i} refers to the farthest destination node in a column, and $1 \leq \hbar \leq 2k$.

Equations (7)-(13) are valid when the network is under light traffic. Most multicast algorithms, including our proposed *QG* algorithm, are obtained from heuristics whose behaviors are difficult to describe using analytical tools, especially in the presence of message contention due to heavy traffic. As a result, a simulation approach is adopted to study the performance of the *QG* algorithm and which is then compared to that of some of the *DP*, *MP*, and *CP* algorithms under various traffic conditions (including the case of light traffic as predicted by the above equations).

4.2 Simulation Experiments

Extensive simulation experiments have been conducted to analyze the performance of *QG* against *DP*, *MP*, and *CP*. A simulation program has been developed to model the multicast operation in the mesh. The developed model has been added to a larger simulator called MultiSim [6], which has been designed to study the collective communication operations on multicomputers and has widely been used in the literature [2], [10], [12]. The simulation program was written in VC++ and built on top of the event-driven CSIM-package [7].

In the simulator, processes are used to model the active entities of the system, which are executed in a parallel fashion, providing a convenient interface for writing modular simulation programs. The main program activates a set of CSIM processes, called multicast generators, one for each network node. Each multicast message is simulated with a pseudoprocess that sends messages to the destinations by creating path pseudoprocesses. A routing model for each is used as path processes to determine the channels on which each message should be transmitted. A statistics module gathers information using the batch mean method [3], [10]. We will present below performance results for the two-dimensional mesh. Other higher dimensional versions of the mesh (e.g., 3D and 4D) are not included. This is mainly due to first the simplicity and space limitation, and second, the excessive computing resources required to run simulations of multicast communication on higher dimensional meshes. In what follows, the two-dimensional mesh will be referred to as simply the 2D mesh throughout our discussion.

A number of simulation experiments have been carried out in order to determine an adequate value for the threshold *TD* $(0 < TD < 1)$ used during the grouping phase in the *QG* algorithm, as discussed in Section 3. As is expected, the multicast latency has been found to be sensitive to the value of the threshold. Decreasing the threshold causes the sizes of the qualified groups to decrease, which, in turn, results in multicast messages traversing short network paths to reach the destination nodes. However, this advantage is achieved at the expense of longer preparation times and more traffic being generated inside the network as a copy of the multicast message has to be sent to each qualified group. On the other hand, as the threshold increases, this causes the grouping process to terminate earlier resulting in larger qualified groups. Although dividing the destinations into large groups leads to fewer messages inside the network, messages traverse longer paths during the multicast operation. Our simulation experiments have revealed that a threshold value of 0.5 is a good compromise as it balances better the performance effects of the group size, preparation time, and generated traffic. Below this value such as 0.2, it enables the partitioning process to create a large number of groups, which will result in longer preparation times and more traffic load. In contrast, the selection of above the 0.5 threshold value such as 0.8 results in larger and unbalanced groups. Hence, 0.5 threshold value has been used in all the simulation results reported in the subsequent sections. For the sake of comparison, our simulation experiments have employed the same assumptions used in previous similar studies [8], [10], [15], which are summarized below. Messages are transmitted from one node to the next using wormhole switching. Each input channel into a router has a single flit buffer size. As the existing studies [8], [10], [15], the 2D mesh with four injection channels and four ejection channels has been used in the experiments.

Messages select network paths according to dimension-ordered routing to cross from source to destination. A source node is randomly chosen and the destination nodes for a given multicast operation are uniformly distributed over the network and each node generates messages independently of any other node and the intergeneration times follow an exponential distribution.

The message length L is fixed (e.g., $L = 32, 64,$ or 128 flits). Each flit requires one cycle transmission time across a channel. Generated messages are queued (based on the FCFS policy) in an infinite capacity queue at the source node. Messages arriving at a destination node are immediately transferred (or ejected from the network) to the local node. The start-up latency (or start-up for short) that accounts for upper layer software overheads (e.g., OS calls) is set at $T_s = 33$ cycles, like [14], [15]. It is worth pointing out that the values used for the start-up latency and channel transmission time have also been used in the studies of [14], [15]. As argued in [10], [14], [15], such values have been adopted because they reflect the characteristics of practical systems such as the CRAY T3D.

The preparation time (which consists of dividing the destination nodes into appropriate subsets and creating multiple sorted lists carrying copies of the message as needed, depending on the algorithm) of *DP*, *MP*, *CP*, and *QG* is set at 2, 2, 4, and 16 cycles, respectively, based on the time complexity of the algorithms, which were discussed before. The preparation time has been deliberately set higher in *QG* to reflect the fact that this algorithm requires a longer time to divide the destinations into groups. It is worth noting that we have found that the results do not change substantially even if this delay factor was set as high as 50 cycles for *QG*. Compared to *DP* and *MP*, both *QG* and *CP* have relatively more complicated procedures to divide the destination nodes into groups. Unlike *DP*, *MP*, and *CP*, the preparation time of *QG* is distributed over a number of nodes contained within the multicast area. Specifically, the preparation time required by the selected nodes in each qualified group takes place simultaneously to reduce the overhead consumed by the preparation phase. In the simulations below, we will be mainly reporting results on the mean multicast latency and maximum throughput. Most previous studies [10], [14], [15], as well as our present study, are interested in the mean latency rather than its exact distribution for the following reason. Many performance studies have revealed [2], [10], [14], [15], [16] that when the input traffic follows a Poisson process and messages are not lost in the event of congestion; the network exhibits long-term steady study behavior as long as it is operating below the saturation point, which is ultimately dictated by the available network bandwidth. In the steady-state region, the mean is usually a good estimate of performance as the coefficient of variation is usually low, implying that most latency values cluster around the mean. This is particularly true when the traffic is low and moderate. When the traffic is heavy causing the network to operate near the saturation point, latency increases and so does the coefficient of variation. In the saturation region, we will be mainly interested in the maximum throughput (or throughput for short) achieved by the network; since in the steady-state region, the maximum throughput is simply equal to the total injected input traffic since there is no message loss. All simulations were executed using 95 percent confidence intervals (when confidence interval was smaller than 5 percent of the mean). The technique used to calculate confidence intervals is called batch means analysis. In batch means method, a long run is divided into a set of fixed size batches, computing a separate sample mean for each batch, and using these batches to compute the grand mean and the confidence interval. In our simulations, the grand means are obtained along with several values, including confidence interval and relative errors as shown in Table 3, which outlines the results depicted in Figs. 4 and 1a, for 10 destinations. However, like existing studies [1], [2], [3], [10, [15], [13], [19], only the grand mean is shown in our figures.

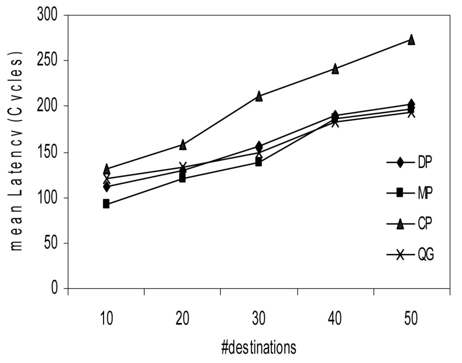
TABLE 3
The Mean and Confidence Interval for the 10 Destinations Shown in Fig. 10

Algorithm	DP	MP	CP	QG
Confidence interval	[107.16-118.08]	[87.62-96.84]	[124.50-136.90]	[114.81-126.71]
Mean	112.621	92.23	130.701	120.760
Relative error	0.05096	0.05263	0.05247	0.05259

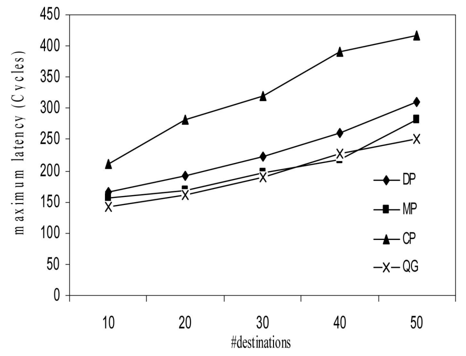
4.2.1 Latency in the Absence of Contention

A number of simulation experiments have been conducted under contention-free conditions in order to assess how well the four algorithms perform under light traffic and also to provide a basis for comparison with the results already reported in the literature, e.g., [10], [15]. One source node is chosen randomly for a multicast operation and destinations, varying from 10, 20... to 50 nodes, are uniformly distributed over the network. The performance metrics used in this set of experiments are the mean multicast latency, maximum multicast latency, and average additional traffic. The mean multicast latency is the average time a multicast operation takes to complete, while the maximum multicast latency is the maximum value recorded among the performed multicast operations in a given simulation run.

The additional traffic is computed as in [8], [10], that is, by subtracting the number of destination nodes from the number of channels involved in the multicast operation. This reflects the amount of network resources that are used to complete a multicast operation. A physical channel occupied for one cycle is considered as one-traffic unit. Fig. 10 compares the mean and maximum multicast latency in the four multicast algorithms against the number of destinations when the message length is set at 32 flits. When the number of destinations is low, *DP* and *MP* outperform *QG* and *CP*. This is because that both *DP* and *MP* require fewer start-ups, namely one start-up. It may be concluded from Figs. 10 and 11 that *DP* and *MP* outperform *QG* under contention-free conditions. However, a potential

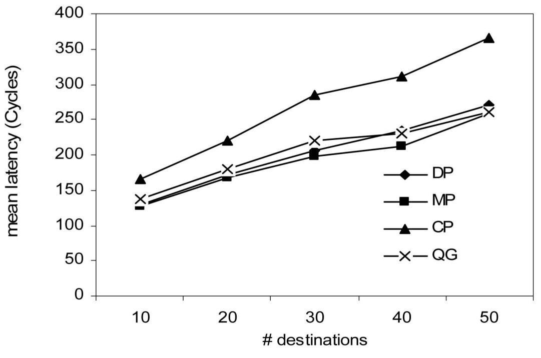


(a)

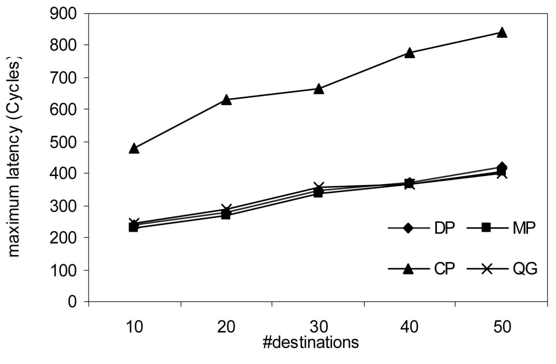


(b)

Fig. 10. (a) Mean latency and (b) maximum latency in the 16×16 mesh with 32 flits message lengths.



(a)



(b)

Fig. 11. Multicast latency in QG, DP, MP, and CP algorithms against the number of destinations in the 16×16 mesh. (a) Mean latency, (b) maximum latency, and message $\text{length} = 128$ flits.

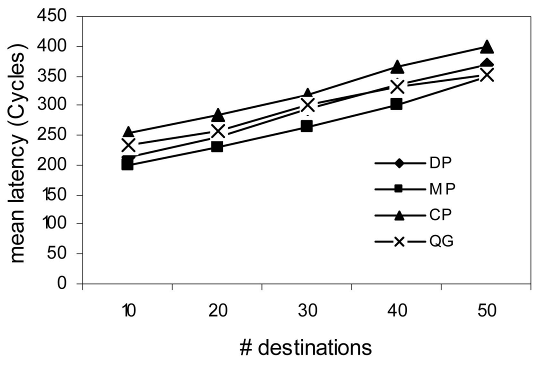


Fig. 12. Mean multicast latency in the QG, DP, MP, and CP algorithms against the number of destinations in the 24×24 mesh. Message $\text{length} = 64$ flits.

disadvantage of *DP* and *MP* is not revealed until either the network size is large or the traffic load is relatively high. Figs. 12 and 13 compare the multicast algorithms when a larger mesh with 24×24 and 32×32 nodes is used, respectively. With a larger set of destinations, the advantage of the *QG* becomes more noticeable as the multicast latency in both *DP* and *MP* becomes higher. This is a result of *QG* generating shorter paths to reach all the destinations.

Even though *DP* and *MP* need fewer start-ups, messages in these two algorithms need to traverse almost the whole network, resulting in higher latencies. *CP* performs poorly because messages experience a higher number of start-ups, especially when the number of destinations is large. Fig. 14 shows the resulting average additional traffic in the four algorithms for various numbers of destination nodes. To complete a multicast operation, *QG* requires fewer channels than *DP*, *MP*, and *CP* since the destinations are divided into several groups, which are reached in a more efficient manner.

4.2.2 Latency at Node Level

This section presents the coefficient of variation of the multicast latency as a new performance metric in order to reflect the degree of parallelism. A set of simulation experiments has been conducted, where the message interarrival time between two messages generated at a source node is set at 250 cycles. The message length is fixed at 64 flits and the number of destination nodes is varied from 20, 30, $40\ldots$ to 80 nodes. The coefficient of variation

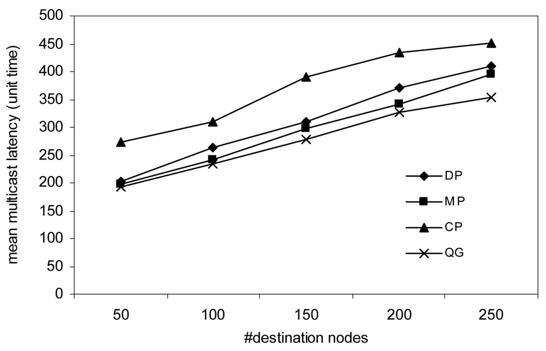
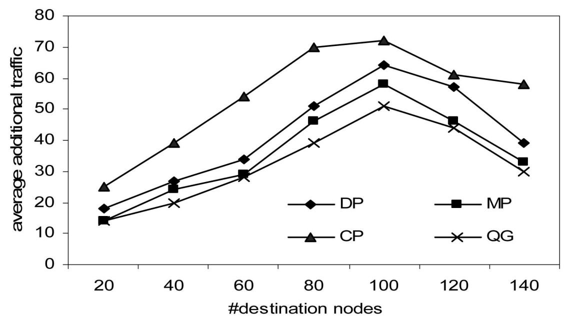
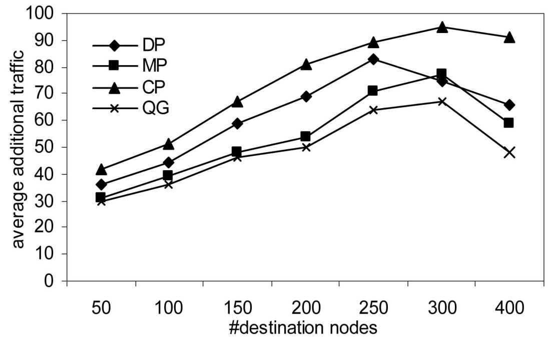


Fig. 13. Multicast latency in QG, DP, MP, and CP algorithms in the 32×32 mesh. Message $\text{length} = 64$ flits.



(a)



(b)

Fig. 14. Average additional traffic as a function of the number of destinations for the four algorithms, DP, MP, CP, and QG: (a) 16×16 mesh, (b) 32×32 mesh.

(CV) is defined as SD/M_{nl}, where SD refers to the standard deviation of the multicast latency (which is also the message arrival times among the destination nodes) and M_{nl} is the mean multicast latency.

The coefficient of variation of *QG* has been compared against that of *DP*, *MP*, and *CP*. Table 4 contains performance results for the 16×16 mesh, which have been obtained by averaging values obtained from at least 40 experiments in each case; the corresponding latency results are reported in Fig. 11. The QG_{IMPR} $percent$ in Table 4 refers to the percentage improvement obtained by *QG* over its *DP*, *MP,* and *CP* competitors. As shown in Table 4, *QG* achieves a significant improvement over *DP*, *MP,* and *CP*. This is due to first the grouping scheme adopted by *QG,* which divides the destinations into groups of comparable sizes. Second, and more importantly, unlike in *DP*, *MP,* and *CP*, the destination nodes for each qualified group in *QG* (except those selected in the first message passing step) receive the multicast message in the second message passing step, in parallel. This has the net effect of minimizing the variance of the arrival times at the node level. In contrast, *DP*, *MP*, and *CP* perform multicast with either longer paths as in *DP* and *MP* or in an excessive number of message passing steps, as in *CP*. Table 5 includes results when the network size is increased to 32×32 nodes; the corresponding latency results are reported in the above Fig. 13. The improvement made by *QG* becomes even more apparent for large system sizes, where *QG* performs the multicast operation with more consideration paid to both the network and node levels while propagating the multicast message. Unlike *DP*, *MP,* and *CP,* which perform the multicast operation in a highly sequential manner and do not scale up well with either the network sizes or the number of destinations, *QG* distributes the load more efficiently so that most destination nodes receive the message in comparable arrival times.

4.2.3 Latency in the Presence of Multiple Multicasts

A wide range of traffic loads have been considered to evaluate the performance of the four multicast algorithms under different working conditions. Two network sizes have been considered, notably 10×10 and 16×16 nodes; network sizes have been limited due to the excessive computing time required to run simulation for larger systems (which require several days, depending on the parameters used). The number of destinations has been set to 10 and 20 and the message size at 64 flits. Figs. 15, 16, and 17 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 multicast communication latency. As a result, *DP* and *MP* outperform their *QG* and *CP* counterparts, due to their fewer start-ups. However, *CP* and *QG* are less sensitive to the increased load than *DP* and *MP*. This is because the former exhibits less traffic in the network as they use shorter paths; so resources are held for shorter time periods, leading to higher throughput (the number of messages successfully delivered to the destination nodes). Figs. 15a and 16a show that *MP* offers a slight improvement over *DP*. This is likely because a message in *MP* reaches destination nodes with shorter paths. However, in high traffic conditions or with a larger number of destination nodes (20 destinations), *QG* has better performance than *DP*, *MP,* and *CP*. Such scenarios are

TABLE 4
The Coefficient of Variation of the Multicast Latency in the DP, MP, and CP Algorithms with the Improvement Obtained by QG (QG_{IMPR} $percent$) in 16×16 Mesh

	# DESTINATIONS=20		# DESTINATIONS =40		# DESTINATIONS =60		# DESTINATIONS =80	
	CV	(QG_{IMPR} %)	CV	(QG_{IMPR} %)	CV	(QG_{IMPR} %)	CV	(QG_{IMPR} %)
DP	0.386	46.19	0.416	54.83	0.476	76.27	0.521	88.24
MP	0.326	23.48	0.365	35.69	0.420	55.56	0.441	59.93
CP	0.467	76.74	0.489	81.56	0.504	86.49	0.549	98.82
QG	CV= 0.2640		CV= 0.2695		CV= 0.27004		CV = 0.276	

TABLE 5
The Coefficient of Variation of the Multicast Latency of the DP, MP, and CP Algorithms with the Improvement Obtained by QG ($QG_{IMPR}\ percent$) in 32×32 Mesh

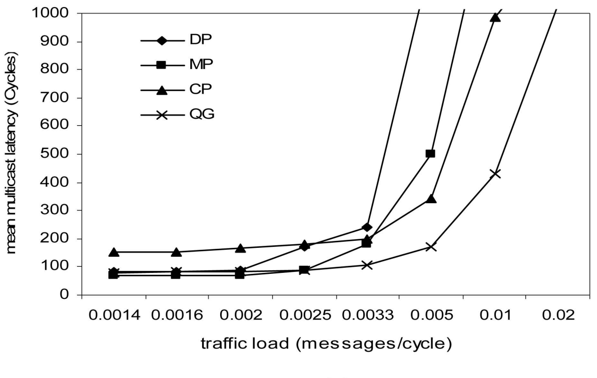
	# DESTINATIONS=20		# DESTINATIONS=40		# DESTINATIONS=60		# DESTINATIONS=80	
	CV	(QG_{IMPR}%)	CV	(QG_{IMPR}%)	CV	(QG_{IMPR}%)	CV	(QG_{IMPR}%)
DP	0.476	58.14	0.531	64.91	0.596	77.44	0.658	94.12
MP	0.441	46.51	0.494	53.44	0.536	59.76	0.577	70.21
CP	0.554	84.04	0.627	94.72	0.680	102.33	0.696	105.3
QG	CV= 0.3009		CV= 0.3219		CV= 0.33588		CV= 0.3389	

illustrated in Figs. 15b and 16b. The same conclusion is obtained even when the preparation time is set at 45 cycles in *QG* as depicted in Figs. 17 and 18. The reason is that the *QG* makes the source node responsible for the multicast operation for a shorter time period than the other algorithms.

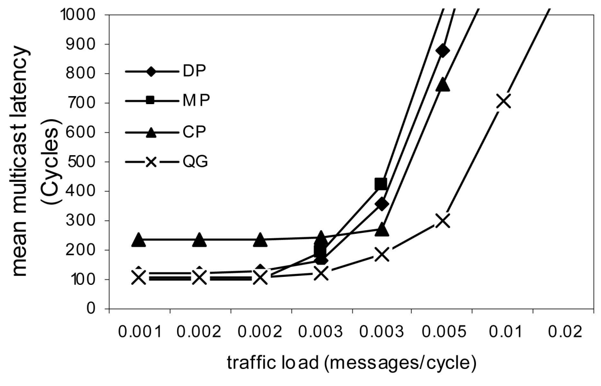
In *QG*, the multicast period involves two different stages. In the first stage, the source node is responsible only for serving the nearest selected nodes of the destination qualified groups. The source node then becomes no longer responsible for the multicast operation since the selected nodes in the qualified groups take over the completion of the multicast operation in their own destination groups, leading to short multicast periods. However, in *DP*, *MP*, and *CP*, the source node remains engaged till the completion of the multicast operation. This engagement has a considerable degrading effect on the network performance as a source node uses its outgoing channels to send out copies of the multicast message. Until the multicast transmission is complete, flits from other messages that route through that source node are blocked at that point. As a consequence, the source node may become a congested point or a hot spot in the network. When the load is high, the source node may throttle system throughput considerably, resulting in a significant increase in multicast latency.

4.2.4 Latency in the Presence of Multicast and Unicast

In some real parallel applications, a message may have to compete for network resources with other multicast messages or even with other unicast messages. To examine performance in such situations, results for the mean multicast latency have been gathered in the 10×10 mesh

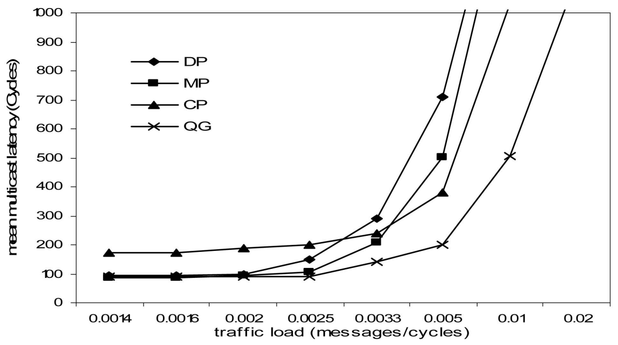


(a)

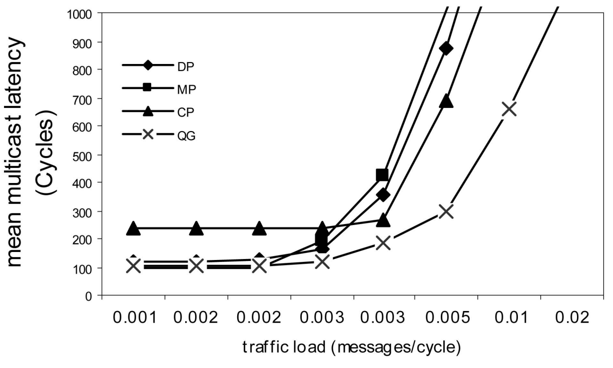


(b)

Fig. 15. Mean multicast latency under different loads in a 10×10 mesh. (a) Number of $\text{destinations} = 10$ nodes, (b) number of $\text{destinations} = 20$ nodes.



(a)



(b)

Fig. 16. Mean latency under different loads in a 16×16 mesh with (a) 10 destinations and (b) 20 destinations.

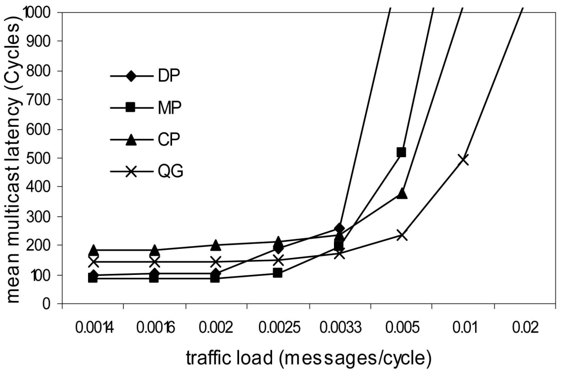


Fig. 17. Mean multicast latency in the QG, DP, MP, and CP algorithms under different traffic loads. 10×10 mesh, number of $\text{destinations} = 10$ nodes. The preparation time is 45 cycles in QG and 2, 2, 8, and 16 cycles in DP, MP, and CP, respectively.

in the presence of both multicast (10 percent) and unicast (90 percent) traffic, (similar studies are outlined in [10], [15]). The message size is set at 64 flits and the number of destinations in a given multicast operation has been set to 10 and 20 nodes, respectively. The simulation results are provided in Figs. 19 and 20; Fig. 19 reports results for 10 destinations while Fig. 20 shows results for 20 destinations. Under light traffic, *QG*, *DP,* and *MP* have comparable performance behavior, with *MP* having a slightly lower latency.

On the other hand, *CP* has a higher time. This is mainly due to the dominating effect of the start-up latency in such a situation. However, under heavy traffic, an opposite behavior is noticed in that *QG* performs the best in terms of both latency and throughput, followed by *CP*. More importantly, we can observe from Fig. 20 that as the number of destinations increases, the performance advantage of *QG* becomes more noticeable over that of *CP*. This is mainly because *QG* alleviates significantly the congestion problem at the source node. In contrast, the source node in *CP* suffers from a higher load and as more destinations are involved in the multicast operation, the more severe this limitation becomes.

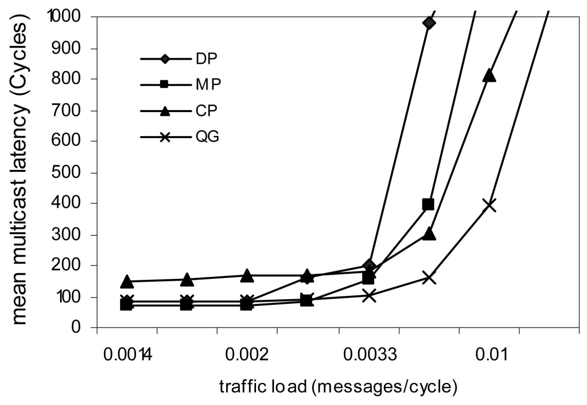


Fig. 19. Mean multicast latency in the 10×10 mesh. Message length is 64 flits, number of 10 $\text{destinations} = 10$ nodes, traffic consists of multicast (10 percent) and unicast (90 percent).

5 Conclusion

This paper has investigated, using extensive simulation experiments, the relative performance merits of *QG* algorithm against three well-known existing algorithms, notably *DP*, *MP* [10], and *CP* [15] under various operating traffic conditions, in the presence of multiple multicast operations and mixture of multicast and unicast messages. Despite the capability of the *QG* to handle multicast in other topologies, such as tours, the target system in this study has been the mesh. The results have shown that in most of the cases considered, the proposed new algorithm has a lower multicast latency and a higher throughput than *DP*, *MP*, and *CP*. The only exceptional case is multicasting in small network sizes under light traffic load (only one single source node at a time). In this situation, *DP* and *MP* perform better than our proposed *QG*. This is due to the fact that in such a situation, the start-up latency is the dominating factor and both *DP* and *MP,* in turn, require fewer start-ups, namely one start-up. However, when a larger mesh used such as 24×24 and 32×32, the advantage of the *QG* becomes more noticeable as the multicast latency in both *DP* and *MP* becomes higher. This is because that *QG* generates shorter paths to reach all the destinations. Even though *DP* and *MP* need fewer start-ups, messages in these

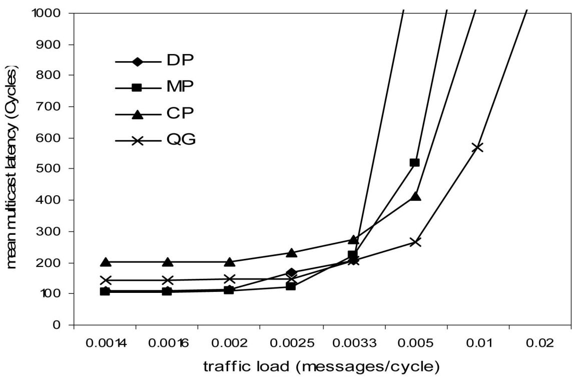


Fig. 18. Mean multicast latency in the QG, DP, MP, and CP algorithms under different traffic loads. 16×16 mesh, number of $\text{destinations} = 10$ nodes. The preparation time is 45 cycles in QG and 2, 2, 8, and 16 cycles in DP, MP, and CP, respectively.

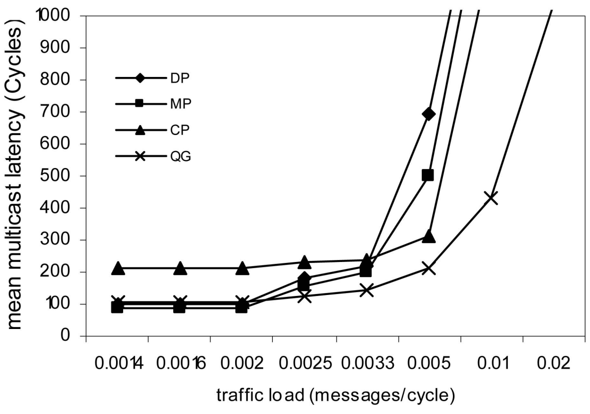


Fig. 20. Mean multicast latency in the 10×10 mesh. Message length is 64 flits, number of $\text{destination} = 20$ nodes, traffic consists of multicast (10 percent) and unicast (90 percent).

two algorithms need to traverse almost the whole network, resulting in higher latencies. The *QG* algorithm exhibits the best performance in terms of other metrics, such as the average additional traffic and latency in the presence of multiple multicast operations with a large number of destinations. The superiority of the *QG* algorithm is due mainly to the ability of its grouping scheme to strike a balance between the performance impact of the start-up latency and the generated traffic inside the network.

Unlike existing works, the *QG* algorithm considers the multicast latency at both the network and node levels (jitter) to achieve a high degree of parallelism during the propagation of a multicast message. The coefficient of variation in the message arrival times at the destination nodes has been used as a performance measure in the simulation experiments to assess the multicast latency of the *QG* algorithm against that of its *DP*, *MP*, and *CP* counterparts at both the network and node levels. The results have revealed that the new algorithm has a lower coefficient of variation. For instance, more than 50 percent improvement has been obtained for small and moderate number of destinations, e.g., 20 to 40 nodes, and more than 90 percent has been achieved for a large number of destinations, e.g., 80 nodes. It would be interesting to further investigate the interaction between the important parameters that affect the performance of the *QG* algorithm, notably the grouping scheme, network size, threshold value, multicast group size, and traffic load, with the aim of proposing an analytical model that could predict, for example, the multicast latency given a particular grouping scheme, network size, multicast group size, and traffic load. For further research, we anticipate that *QG* scheme can be implemented in different network systems, including network-on-chip, which could provide a concrete basis for a number of interesting extensions.

Acknowledgments

The authors would like to thank the anonymous reviewers for the insightful and constructive comments on their manuscript.

References

[1] N.-C. Wang, C.-P. Yen, and C.-P. Chu, "Multicast Communication in Wormhole-Routed Symmetric Networks with Hamiltonian Cycle Model," *J. Systems Architecture,* vol. 51, no. 3, pp. 165-183, Mar. 2005.

[2] A. Al-Dubai, M. Ould-Khaoua, and L. Mackenzie, "An Efficient Path-Based Multicast Algorithm for Mesh Networks," *Proc. Int'l Parallel and Distributed Processing Symp. (IPDPS),* p. 283, 2003.

[3] J. Duato, C. Yalamanchili, and L. Ni, *Interconnection Networks: An Engineering Approach.* Elsevier Science, 2003.

[4] A. Touzene, "Optimal All-Ports Collective Communication Algorithms for the *k*-Ary *n*-Cube Interconnection Networks," *J. Systems Architecture,* vol. 50, no. 4, pp. 169-236, 2004.

[5] C. Busch, M. Magdon-Ismail, and J. Xi, "Optimal Oblivious Path Selection on the Mesh," *IEEE Trans. Computers,* vol. 57, no. 5, pp. 660-671, May 2008.

[6] P.K. McKinley and C. Trefftz, "MultiSim: A Simulation Tool for the Study of Large-Scale Multiprocessors," *Proc. Int'l Symp. Modeling, Analysis and Simulation of Computer and Telecomm. Systems (MASCOTS '93),* pp. 57-62, 1993.

[7] CSIM: Internet: http://www.mesquite.com/, 2009.

[8] E. Fleury and P. Fraigniaud, "Strategies for Path-Based Multicasting in Wormhole-Routed Meshes," *J. Parallel and Distributed Computing,* vol. 60, pp. 26-62, 1998.

[9] D. Xiang, Y. Zhang, and Y. Pan, "Practical Deadlock-Free Fault-Tolerant Routing in Meshes Based on the Planar Network Fault Model," *IEEE Trans. Computers,* vol. 58, no. 5, pp. 620-633, May 2009.

[10] X. Lin, P. McKinley, and L.M. Ni, "Deadlock-Free Multicast Wormhole Routing in 2D-Mesh Multicomputers," *IEEE Trans. Parallel and Distributed Systems,* vol. 5, no. 8, pp. 793-804, Aug. 1994.

[11] S. Cang and J. Wu, "Time-Step Optimal Broadcasting in 3-D Meshes with Minimal Total Communication Distance," *J. Parallel and Distributed Computing,* vol. 60, pp. 966-997, 2000.

[12] D.F. Robinson, P.K. McKinley, and C. Cheng, "Path Based Multicast Communication in Wormhole Routed Unidirectional Torus Networks," *J. Parallel Distributed Computing,* vol. 45, pp. 104-121, 1997.

[13] A. Agarwal, "Limits on Interconnection Network Performance," *IEEE Trans. Parallel and Distributed Systems,* vol. 2, no. 4, pp. 398-412, Oct. 1991.

[14] P. Mohapatra and V. Varavithya, "A Hardware Multicast Routing Algorithm for Two Dimensional Meshes," *Proc. Eighth IEEE Symp. Parallel and Distributed Processing,* pp. 198-205, Oct. 1996.

[15] R.V. Boppana, S. Chalasani, and C.S. Raghavendra, "Resource Deadlock and Performance of Wormhole Multicast Routing Algorithms," *IEEE Trans. Parallel and Distributed Systems,* vol. 9, no. 6, pp. 535-549, June 1998.

[16] S. Wang, Y. Tseng, C. Shiu, and J. Sheu, "Balancing Traffic Load for Multi-Node Multicast in a Wormhole 2D Torus/Mesh," *The Computer J.,* vol. 44, no. 5, pp. 354-367, 2001.

[17] Z. Lu, "Design and Analysis of On-Chip Communication for Network-on-Chip Platforms," PhD thesis, Royal Inst. of Technology, Sweden, Mar. 2007.

[18] D. Panda, S. Singal, and R. Kesavan, "Multidestination Message Passing in Wormhole *k*-Ary *n*-Cube Networks with Base Routing Conformed Paths," *IEEE Trans. Parallel and Distributed Systems,* vol. 10, no. 1, pp. 76-96, Jan. 1999.

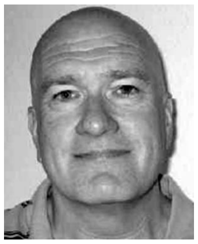
[19] K.-P. Fan and C.-T. King, "Turn Grouping for Multicast in Wormhole-Routed Mesh Networks Supporting the Turn Model," *The J. Supercomputing,* vol. 16, no. 3, pp. 237-260, Jul. 2000.

[20] X. Zhuang and V. Liberatore, "A Recursion-Based Broadcast Paradigm in Wormhole Routed Networks," *IEEE Trans. Parallel and Distributed Systems,* vol. 16, no. 11, pp. 1034-1052, Nov. 2005.

[21] Y. Chang, J. Wu, C. Chen, and C. Chu, "Improved Methods for Distribution on *k*-Dimensional Meshes Using Multi-Installment," *IEEE Trans. Parallel and Distributed Systems,* vol 18, no. 11, pp. 1618-1629, Nov. 2007.

Ahmed Y. Al-Dubai received the BSc and MSc degrees in computer science from Mutah University and Al al-Bayt University, Jordan, in 1996 and 1999, respectively, and the PhD degree in computing from the Department of Computing Science, University of Glasgow, in 2004. He is currently working as a lecturer in the School of Computing at Edinburgh Napier University. He was a full time lecturer at Thames Valley University, London, (2004-2005) before joining Edinburgh Napier University. His research interests include communication algorithms, parallel and distributed computing, and next generation wired and wireless networks. He served on organizing and program committees for many prestigious IEEE and ACM international conferences. He is the recipient of several prestigious awards and recognitions. He has been the guest coeditor of several scholarly journals. He is the vice program chair of the 10th IEEE International Conference on High Performance Computing and Communications (HPCC '08), DaLian, China; the program chair of the 7th and 8th International IEEE/ACM Workshop on Performance Modeling, Evaluation, and Optimization of Ubiquitous Computing and Networked Systems (PMEO-UCNS), held in conjunction with the IEEE IPDPS 2008 and 2009. He is also the cochair of the International Workshop on Wireless Sensor and Ad-hoc Networks (WiSAN '08), held in conjunction with the IEEE Computer Society Press' 37th ICPP 2008, Portland, Oregon. He is a senior member of the IEEE, the IEEE Computer Society, and the ACM.

Mohamed Ould-Khaoua received the BSc degree from the University of Algiers, Algeria, in 1986, and the MAppSci and PhD degrees in computer science from the University of Glasgow, United Kingdom, in 1990 and 1994, respectively. He is currently working as a professor in the Department of Electrical and Computer Engineering, Sultan Qaboos University, Oman. Prior to that, he worked as a lecturer from 2000 to 2003 and then as a reader from 2004 to 2007 at the University of Glasgow, United Kingdom. He also worked as a lecturer at Starthclyde University, United Kingdom, from 1997 to 2000 and as a postdoctoral research fellow at Teesside University, United Kingdom, from 1994 to 1997. He is the editor-in-chief of *The Journal of Engineering Research* (TJER). He serves on the Editorial Board of the *IEEE Transactions on Parallel and Distributed Systems* (IEEE TPDS), *International Journal of Parallel, Distributed and Emergent System* (IJPDES), *International Journal of Computers and Applications* (IJCA), *International Journal of High Performance Computing and Networking* (IJHPCN), *International Review on Computers and Software* (IRECOS), and *International Journal of Computer and System Sciences*. He is the guest editor of more than 19 special issues on topics related to systems and networks performance evaluation. He was a cochair of the international workshop series on performance modeling, evaluation, and optimization of parallel and distributed systems (PMEO-PDS), the ACM International Workshop on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks (PE-WASUN), the ACM International Workshop on Performance Monitoring, Measurement, and Evaluation of Heterogeneous Wireless and Wired Networks (PM2HW2N), the International Workshop on Performance Evaluation of Networks for Parallel, Cluster, and Grid Computing Systems (PEN-PCGCS), and the International Workshop on Performance Modeling and Analysis of Communication in Parallel, Distributed, and Grid Networks (PMAC-PDG). His current research interests are performance modeling and evaluation of wired/wireless networks and parallel/distributed systems. He is a member of the IEEE and the IEEE Computer Society.



Lewis M. Mackenzie received the BSc degree in mathematics and natural philosophy from the University of Glasgow, United Kingdom, in 1980, and the PhD degree in 1984, also from the University of Glasgow, for work on the development of multicomputers for use in nuclear physics. He is currently working as a lecturer at the Department of Computing Science at the University of Glasgow, which he joined in 1985. His current research interests include multicomputers, high-performance networks, mobile ad hoc and vehicular networks, and hypercomputation. He is a member of the IEEE and the IEEE Computer Society.