A Scalable Broadcast Algorithm for Multiport Meshes with Minimum Communication Steps

Ahmed Yassin Al-Dubai and Mohamed Ould-Khaoua

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

Abstract

Many broadcast algorithms have been proposed for the mesh over the past decade. However, most of these algorithms do not exhibit good scalability properties as the network size increases. As a consequence, most existing broadcast algorithms cannot support real-world parallel applications that require large-scale system sizes due to their high computational demands. Motivated by these observations, this study proposes a new adaptive broadcast algorithm for the mesh. The unique feature of our algorithm is that it handles broadcast operations with a fixed number of message-passing steps irrespective of the network size. Our algorithm is based on the coded path routing, which has been proposed in [1]. Results from extensive comparative analysis reveal that the proposed algorithm exhibits superior performance characteristics over those of the well-known Recursive Doubling and Extending Dominating Node algorithms.

1. Introduction

Collective communication, such as broadcast, which refers to the delivery of the same message originating from a given source to all network nodes, is important in many real-world parallel applications found in the areas of Science and Engineering [4, 12]. For instance, broadcast communication is often needed in scientific computations to distribute large data arrays over system nodes in order, for example, to perform various data manipulation operations. Furthermore, it is required in control operations such as global synchronisation and to signal changes in network conditions, e.g., faults. In the distributed shared-memory paradigm, broadcast communication is often used to support shared data invalidation and updating procedures required for cache coherence protocols [12].

Several broadcast algorithms have been proposed in the literature for the wormhole-switched mesh [2, 3, 4, 11, 17]. These algorithms try to reduce the broadcast latency by reducing the number of message-passing steps, i.e. the number of exchanges, required to perform a broadcast operation. However, most of these algorithms do not scale well with the system size as they suffer from the degrading effects of the start-up latency, the required time to handle a broadcast message at both the source and destination nodes [3], especially when the network size is large. This is because the number of message-passing steps that is required to complete a broadcast operation usually depends on the network size. In this paper, the Coded Path Routing (or CPR for short) which has been proposed in [1] will be used as an efficient approach for designing efficient broadcast algorithms for the mesh.

Specifically, the CPR is used to devise a new broadcast algorithm for the multiport 3D meshes. Owing to the properties of the CPR, the proposed algorithm requires a minimal number of message-passing steps to implement a broadcast operation, irrespective of the system size. An extensive comparative analysis presented below reveals that the new broadcast algorithm exhibits superior performance characteristics over the well-known Recursive Doubling and Extending Dominating Node and Network Partitioning algorithms proposed in [2] and [3], respectively.

The remainder of this paper is organised as follows. Section 2 outlines the motivation of this study. Section 3 describes briefly the system model. The new broadcast algorithm for multiport 3D meshes lies in Section 4. Section 5, is devoted to the performance of the proposed algorithm, Recursive Doubling and Extending Dominating Node algorithms. Finally, Section 6 concludes this study.

2. The Problem and Motivation

In general, most existing studies [2, 3, 4, 11] have focused on minimising the number of message-passing steps required for collective communication, such as broadcast. However, there has been hardly any study that has considered minimising the effects of the network size on the performance of broadcast algorithms. As a result, most existing algorithms do not scale well with the network size as the number of message-passing steps
increases proportionally with the system size. In other words, the larger the underlying interconnection network is, the more severe this limitation becomes. As a result, many parallel application cannot be implemented efficiently, such as, real time application and synchronisation, where the processors should receive the broadcast message in comparable message arrivals. In addition, most of the existing broadcast algorithms in the literature [2, 3, 11, 18] handle broadcast with deterministic routing. Hardly any study has exploited the performance advantages of adaptive routing to develop efficient broadcast algorithms. To address this, the present study proposes a new broadcast algorithm that uses an adaptive routing and maintains good performance levels for various system sizes.

Motivated by these observations, this study proposes a new adaptive broadcast algorithm for the mesh. The unique feature of our algorithm is that it handles broadcast operations with only three message-passing steps irrespective of the network size. Our algorithm is based on the coded path routing, which has been proposed in [1]. Results from extensive comparative analysis reveal that the proposed algorithm exhibits superior performance characteristics over those of the well-known Recursive Doubling and Extending Dominating Node algorithms.

3. The System Model

A 3-dimensional mesh has \( N = N_x \times N_y \times N_z \) nodes, arranged in the three dimensions X, Y, and Z, respectively, with \( N_x \), \( N_y \), and \( N_z \) being the number of nodes in the three dimensions. A node is identified by a three co-ordinate vector \((x, y, z)\), where \( 0 \leq x \leq N_x - 1 \), \( 0 \leq y \leq N_y - 1 \), \( 0 \leq z \leq N_z - 1 \). The mesh topology is asymmetric due to the absence of the wrap-around connections along each dimension. As a result, nodes may not be connected to the same number of neighbours; those at the corners, edges, and middle of the network have four and six neighbours respectively. In this system, the node consists of a processing element (PE) and router.

The PE contains a processor and some local memory. There are local channels used by the PE to inject/eject messages to/from the network, respectively. Messages generated by the PE are injected into the network through the injection channel. Similar to the previous studies of [9, 10], this study considers the multiple-port router model where multiple copies of the broadcast message can be injected into the network through different output channels concurrently.

4. A New Broadcast Algorithm

The CPR exploits the main features of wormhole switching, such as few buffer requirements and distance insensitivity, to overcome the limitations of the existing approaches, and to efficiently support collective communications. In the CPR, the header flit has two bits that form the control field. The two bits indicate to a router which action to take, e.g., pass or receive, upon the reception of a message.

Procedure Control Field (message, operation)

Begin

1. receive the second field of the message;
   if (current router is not the addressed router) then
     if (control field = 10) then pass the message to the next router;
     else
       if (control field = 01) then receive the message;
       else
         if (control field = 11) then
           receive the message; pass the message to the next router;
         else receive the message;
   End

Figure 1. The “Control Field” algorithm used in the CPR.

Fig. 1 describes the “Control Field” algorithm that the router uses to either interpret or modify the control field. In fact, the two bits of the control field have originally been specified in order to enable the CPR to be used in different systems, such as those using one-port or multiple port router models, and also to support different types of collective communication operations, including broadcast and multicast. However, to illustrate the advantages of the CPR, we will focus our discussion in the present study on the use of the CPR for the development of broadcast algorithms; we plan to extend in the future the application of the CPR to multicast communication.

As in the path-based algorithms of [12], which use the multidestination approach, it is assumed that a router in the CPR can simultaneously receive a message and passes a copy to the next router. Due to space limitation, we refer the reader to [1] for more detail on the CPR. While most previous broadcast algorithms for meshes have been discussed in the context of deterministic routing [2, 3, 11, 18], this section introduces the “Double Sided” (or DS for short) algorithm as a new broadcast algorithm that is based on adaptive routing. The DS uses the Turn model discussed in [5] to achieve routing adaptivity while ensuring deadlock freedom (due to space limitation, we will omit the description of this routing algorithm).
algorithm. We refer the reader to [5] for more detail). While the Turn model prohibits just enough turns to ensure deadlock freedom, its adaptivity feature provides the DS algorithm with a greater flexibility in choosing a network path for a message during a given message-passing step. Fig. 2 describes the broadcast operation in the DS algorithm. The proposed algorithm exploits the features of the CPR to implement broadcast in three message-passing steps, thus considerably reducing the effects of both the network size and start-up latency. Examining Fig. 2 reveals that the proposed algorithm achieves a highly degree of parallelism during the propagation of the broadcast message from one router to the next. This has the net effect of greatly reducing the overall time required to complete a broadcast operation.

As a first step and from any starting node, route a message to the nearest corner of the nearest side of the mesh, and also to a corner of the opposite side. For instance, if the side $+X$ has been chosen, the opposite side will be $-X$. In this step, the value of the control field is set at 10. Then, the selected corner acts as a source node and broadcasts the message in its own side after changing the control field to 11. In other words, route a single message through all of the nodes in sequence on each of these two sides. Finally, route messages from these faces into the interior of the mesh following paths perpendicular to the sides, covering all remaining mesh nodes.

The “Double Sided” Broadcast Algorithm

/* Input: source node $(S_x, S_y, S_z)$; $M$: Message; Output: All nodes receive a copy of $M$ */
{
Control field := 00; Check the source node location; Change the value of the control field to “10”;
Select two sides $R_1, R_2$ $(+X, -X, +Y, -Y, +Z, -Z)$;
Let $R$ be the selected dimension of the two selected sides;
Select the nearest two corners $c_1, c_2$ to the source node $(S_x, S_y, S_z)$;
Send out the $M$ to the selected corners in parallel;
forall $c_1, c_2$ in parallel do
{
Change the value of the control field to 11;
Send out the $M$ to $R_1, R_2$ respectively;
}
forall $(x, y, z) \in R_1$ and $(x, y, z) \in R_2$ in parallel do
{
if $(x, y, z) \in R_1$ then
Send out the $M$ to the opposite node that has the value $\frac{1}{2}R_1$ on the $R$;
else
Send out the $M$ to the opposite node that has the value $(\frac{1}{2}R_2 - 1)$ on the $R$;
}
}

Fig 2: A simple description of the proposed “Double Sided” broadcast algorithm

5. Performance and Comparison

This subsection reports on the performance of the DS under dynamic situations using simulation experiments. A simulation program was used to model the broadcast operations of the DS in the 3D mesh. The program was written in VC++ and built on top the event-driven CSIM 18-package [16]. In the simulator, processes are used to model the active entities of a system, and can execute in a quasi-parallel fashion, providing a convenient interface for writing modular simulation program. In our case, every node is modelled as a process.
For studying broadcast operation, the main program activates a set of CSIM parallel processes that are used to broadcast a message in the network. In each experiment, different source nodes were chosen randomly using a uniform number generator. Each broadcast message is simulated with pseudo-process that sends the 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. Each channel has a single queue of messages waiting for transmission. The statistics were collected with 95% confidence interval when the system reaches the steady state; when results do not change with time. The message length was varied from 32 to 2048 flits and the channel rate was set at $\beta = 0.003 \, \mu s$.

The RD was originally proposed by Barnett et al [2]. This algorithm requires $\log_2 N$ steps for broadcasting in 3D mesh. In this algorithm, each node holding a copy of the message is responsible for a partition of a row or column, which will be then divided in half. In each half, a node sends a copy of the message to the node in the other half that occupies the same relative position. This process is implemented recursively until the completion of the broadcast operation. In the absence of contention problem, it can fully take advantage of the pipelining effect of wormhole switching. The EDN was proposed by Tsai and McKinley [3].

The EDN was proposed by Tsai and McKinley [3], and can systematically construct collective operations in multiport wormhole-routed networks. In the EDN approach, the network is divided into several levels. For each level, a dominating set is assigned. For instance, a dominating set $D$ of a graph $G$ is a set of vertices in $G$ such that every vertex in $G$ is either in $D$ or is adjacent to at least one vertex in $D$. However, the EDN requires that the number of nodes along a given dimension to be multiple of 4. The authors in [20] have shown that the number of message-passing steps required in a network size such as $(4 \times 2^k)(4 \times 2^k)(4 \times 2^m)$ or $(4 \times 2^k)(4 \times 2^k)(4 \times 3^m)$ is $k + m + 4$, where $k$ and $m \geq 0$. In addition, even though the EDN can take the advantages of the multiport router model (we assume that the EDN uses three ports router in the present study), it requires a number of messages-passing steps increasing proportionally with the system size.

We assumed system parameters representing the current trends in technology, i.e., two different values for the start-up latency of $1.5 \, \mu s$ and $0.15 \, \mu s$ were considered in Figs. 4 and 5, respectively.

The communication latency of the broadcast operation against the message length has been considered in our experiments. To study the effect of the message length on the communication latency of the three algorithms, different message lengths, $L = 32$ to 2048 flits, have been considered. In Fig. 3, the network size is set at $N = 8 \times 8 \times 8$ nodes; it is worth noting that the same conclusions are obtained when larger network sizes are considered. The start-up latency and channel rate are set at $(\alpha + \gamma) = 1.5 \, \mu s$ and $\beta = 0.003 \, \mu s$, respectively. The values selected for the parameters start-up latency and channel rate are realistic given the current implementation technology used in practical systems, such as the Cray T3D machine [10].

![Fig. 3: The Performance of the DS, RD and EDN algorithms with short messages. Network size: $8 \times 8 \times 8$ and start-up latency 1.5 \, \mu s.](image)

The results shown in Figs. 3 and 4 reveal that the DS has a noticeable performance advantage over both the RD and EDN. This is because the new algorithm implements broadcasting with a high degree of parallelism. For comparison, Fig. 4 depicts that the RD achieves better performance than the EDN in the case of short message lengths. However, when it comes to the long messages an apposite behaviour is obtained. Examining performance results obtained in Figs. 3 and 4 do confirm the fact that the performance of both the EDN and RD is highly dependent on the message length.

To study the effect of the network size on the performance of the three algorithms, various sizes have been considered including $N = 4 \times 4 \times 4$, $8 \times 8 \times 8$, $16 \times 16 \times 16$ and $64 \times 64 \times 64$ nodes. Figs. 3, 4 and 5 show that both the EDN and RD have a poorer performance than the DS. In Fig. 5, the results reveal that as the network size increases, the performance of the EDN and RD degrades significantly. Therefore, both of the RD and EDN do not match the good scalability of the mesh. In contrast, the DS achieves the highest parallelism during broadcasting operation in all different network sizes. Furthermore, it manages to maintain a good level of performance irrespective of the network size. This is because in the DS, when the network size increases, only the distance (i.e., the number of nodes) traversed by the message increases while there is no increase in the number of message-passing steps required to complete the
broadcast operation. However, both of the EDN and RD implement the broadcast operation in a highly sequential manner, i.e., they require more message-passing steps to implement broadcast operations as the network size increases.

Since each message-passing step requires start-up latency, both the EDN and RD incur a higher communication latency to perform broadcast operations. In terms of the effect of the start-up latency, most existing broadcast algorithms suffer from the degrading effects of the start-up latency.

This effect can be minimised by reducing the number of message-passing steps required to perform a broadcast operation. We have assessed the effects of the start-up latency on the performance of the three broadcast algorithms by considering two different start-up latencies, notably 0.15 \( \mu s \) and 1.5 \( \mu s \). In Figs. 3, 4, the start-up latency was set at 1.5 \( \mu s \) whereas in Fig. 5 it was set at 0.15. In all Figs. 3 to 5, it is concluded that the DS achieves the best performance. This is due to the fact that the DS implements broadcast with only three start-up latencies as it relies on a high degree of concurrent transmissions of messages along the network dimensions (see the outline of the DS algorithm in Fig. 2). In the case of higher start-up latency (1.5 \( \mu s \)), Fig. 5 shows that both of the RD and EDN exhibit logarithmic increasing in communication latency while that of the proposed algorithm, DS does linearly as the network size increases, achieving a considerable degree of scalability. Unlike Fig. 5, Fig. 6 shows that the RD and EDN have a similar performance. However, the RD outperforms the EDN in large network sizes. This is due to the fact that the EDN implements broadcast operation in a sequential manner with more start-up latency than that required for the RD algorithm.

6. Conclusion and Future Directions

While the existing broadcast algorithms are implemented sequentially and, therefore, do not scale well with the network size, this paper has suggested an efficient broadcast algorithm, which overcomes the limitations of the existing algorithms. The proposed algorithm has the main advantage of requiring only three message-passing steps irrespective of the network size.

Unlike the previously proposed algorithms, our algorithm achieve a high degree of parallelism during the propagation of the broadcast message from one router to the next, i.e., most of the network nodes receive the broadcast message in parallel. Moreover, our simulation results have revealed that the proposed algorithm has superior performance characteristics than the existing Recursive Doubling Extending Dominating Node algorithms. The next step in our work is to extend our work towards devising new collective communication algorithms such as multicast 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


