Towards Optimal Broadcasting in Wormhole-Routed Meshes

A. Y. Al-Dubai, M. Ould-Khaoua and L. M. Mackenzie

Department of Computing Science, University of Glasgow, Glasgow G12 8RZ, UK

Abstract: Most existing broadcast algorithms proposed for the mesh do not scale well with the network size. Furthermore, they have been mainly based on deterministic routing, which cannot exploit the alternative paths provided by mesh topology to reduce communication latency. Motivated by these observations, this paper introduces a new adaptive broadcast algorithm for the mesh based on the coded path routing approach. The unique feature of the new algorithm is its ability to handle broadcast operations with only two message-passing steps irrespective of the network size. Results from a comparative analysis reveal that the proposed algorithm exhibits superior performance characteristics over those of the well-known Recursive Doubling, Extending Dominating Node and Network Partitioning algorithms.

Keywords: Mesh, Broadcast, Wormhole Switching, Adaptive Routing, Communication Latency, Performance Analysis.

1. Introduction

Among the various interconnection networks, the mesh has been one of the most common networks for practical multicomputers. This is due to its desirable properties, such as ease of implementation, recursive structure, and ability to exploit communication locality found in many parallel application to reduce message latency. The J-machine, Intel Touchstone Delta, Stanford DASH and M-Machine are examples of practical systems that are based on the mesh topology.

Wormhole switching has also promoted the use of the mesh as it makes latency almost insensitive to the message distance, and also simplifies router design due to its minimal buffering requirement. In wormhole switching, a message is divided into elementary units called flits, each of a few bytes for transmission and flow control. The header flit (containing routing information) governs the route and the remaining data flits follow it in a pipelined fashion. If a channel transmits the header of a message, it must transmit all the remaining flits of the same message before transmitting flits of another message. When the header is blocked the data flits are blocked in-situ.

A routing algorithm is responsible for selecting a path that a message can take to reach its destination, and has great impact on network performance. Routing algorithms can be classified as either deterministic or adaptive. In deterministic routing, messages with the same source and destination addresses always take the same network path. As a result, they cannot take advantage of alternative paths that a topology may provide to avoid blocking, and thus reduce their latency. Many adaptive routing algorithms have been proposed in the literature where a message can use alternative paths between a given pair of nodes to advance through the network [5].

The main limitation of adaptive routing is its requirement for extra hardware resources, e.g., virtual channels, to deal with the problem of message deadlock. Adding virtual channels often translates into high hardware complexity, which can significantly reduce router speed, decreasing the overall network performance [5]. The hardware cost of adaptive routing has motivated researchers to develop adaptive algorithms that can achieve adaptivity without using virtual channels, leading to an efficient router implementation [7, 8]. The Turn model is an example of an adaptive routing that was proposed by Glass and Ni [6] for the mesh. This form of routing of routing does not require virtual channels because it ensures deadlock-freedom by prohibiting certain turns in the network. For instance, there are 4n (n - 1) turns in an n-dimensional mesh. As demonstrated in [6], prohibiting a quarter of the turns is sufficient for preventing message deadlock.

Several broadcast algorithms have been proposed in the literature for the wormhole-switched mesh [2, 3, 15, 16]. These algorithms try to reduce the broadcast latency by reducing the number of message-passing steps, i.e. the number of message exchanges between nodes, 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 [15], 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. Moreover, although the mesh topology provides more than one path between a given pair of nodes, almost all the proposed broadcast algorithms [2, 3, 15, 16] have been based on deterministic routing. Motivated by these observations, this paper introduces a new adaptive broadcast algorithm for the mesh based on the coded path routing approach [1]. Owing to the properties of the CPR, the proposed algorithm requires only two message-passing steps to implement a broadcast operation, irrespective of the system size. Furthermore, an important feature of the new algorithm is the use of adaptive routing to provide a greater flexibility in choosing a path for a broadcast message, compared to deterministic routing. We have opted to use the 2-dimensional mesh (or 2D mesh for short) in our present discussion because most existing studies [2, 3, 15, 16] have focused on the 3D mesh; although higher-dimensional meshes (e.g. 4D and higher) are also interesting from a theoretical viewpoint, they have not been used in practice, and therefore they have been omitted from our discussion. An extensive comparative analysis presented below reveals that the new broadcast algorithm exhibits superior performance characteristics over the well-known Recursive Doubling, Extending Dominating Node and Network Partitioning algorithms of [2], [15] and [16], respectively. The remainder of this paper is organised as follows. Section 2 reviews the existing broadcast algorithms that have been proposed in the literature for multicomputer networks, such as the mesh. Section 3 introduces the CPR. Section 4 presents the System model. The new broadcast algorithm for the mesh lies in section 5. Section 6 compares the performance of the proposed algorithm to the Recursive Doubling and Extending Dominating Node algorithms. Finally, Section 7 concludes this study.

2. Related Work and Motivation

Existing algorithms that have been suggested for collective communication, such as broadcast, are founded on either the unicast-based [10] or multidestination-based approach [8, 12, 13]. The main objectives of the two approaches is firstly to reduce the waste of network bandwidth due to additional traffic caused by excessive replication of broadcast messages inside the network, and secondly to reduce the communication delay due to the start-up latency. The start-up latency varies from one system to another, and is usually higher than channel transmission times in terms of current implementation technology [5].

Current practical multicomputers have adopted the unicast approach due to its simplicity. Collective communication based on this approach is implemented as a sequence of unicast message exchanges as it uses the same routing provided for normal unicast (one-to-one) messages. The algorithms proposed by Barnett et al [2], Tsai and McKinley [15] and Cang and Wu [16] are examples that use the unicast approach. Unfortunately, these algorithms use several phases of message exchanges, each phase encountering separated start-up latency. Consequently, the communication overhead can be significant and detrimental to network performance, especially in the presence of a high start-up latency [18].

Several researchers have used the multidestination approach to reduce the degrading effects of the start-up latency, [8, 12, 13]. A message in this approach can carry many addresses, and can be delivered to a group of destinations in a single message-passing step. The source node generates an ordered list of destinations, i.e. depending on the intended order of traversal, and incorporates it into the header flit. Unfortunately, the multidestination approach suffers from several limitations. Firstly, Malumbres and Duato [9] have shown that each message-passing step requires a message preparation phase to sort \( n \) addresses with a minimum software cost of \( O(n \times \log n) \). Consequently, this preparation phase may take more time than the actual message transmission time especially when \( n \) is high. Secondly, since the list of addresses in the header flits are sorted, the routing may not use a minimal path for all of the sorted destinations. This increases the message journey through the network, and as a result may lead to increased message contention inside the network. Thirdly, due to the presence of many addresses in the header, each address occupies one flit. If each flit in the header is assumed to require one updating cycle, this approach requires \( (n - 1) \) additional communication cycles that are spent in updating \( n \) addresses in the header flits. Therefore, several researchers attempted to alleviate the drawback of multidestination-based approach.

Existing studies [2, 3, 15, 16] 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 addition, all existing broadcast algorithms in the literature [2, 3, 15, 16] 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 adaptive routing and maintains good performance levels for various system sizes.

3. Coded Path Routing (CPR)

This section presents the Coded Path Routing (CPR) approach that can reduce the overhead due to the start-up latency and the effects of the network size on the performance of collective communication operations. The CPR exploits the main features of wormhole switching, such as its low buffer requirement and distance insensitivity, to efficiently support collective communication. 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. We refer the reader to [1] for more detail on the CPR.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Code</th>
<th>Pass</th>
<th>Receive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stop a message “default”</td>
<td>00</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Receive a message</td>
<td>01</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Pass a message</td>
<td>10</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Receive and Pass a message</td>
<td>11</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 1 summarises the possible operations on 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.

4. The System Model

A 2-dimensional mesh has \( N = N_x \times N_y \) nodes, arranged in the two dimensions \( X \) and \( Y \), respectively, with \( N_x \) and \( N_y \) being the number of nodes in the two dimensions. A node is identified by a two coordinate vector \((x, y)\), \( 0 \leq x \leq N_x - 1 \), \( 0 \leq y \leq N_y - 1 \). The mesh topology is asymmetric due to the absence of the wrap-around connections along each dimension. Therefore, nodes may not be connected to the same number of neighbours; those at the corners, edges, and middle of the network have 2, 3 and 4 neighbours respectively. There are four corners in the mesh are located at the following addresses: \((N_x - 1, N_y - 1)\), \((N_x - 1, 0)\), \((0, N_y - 1)\), \((0, 0)\).

A node consists of a processing element (PE) and router. The PE contains a processor and some local memory. The router has four inputs and four output channels. A node uses four inputs and four output channels to connect to its neighbouring nodes; two in a dimension, one for each direction. There are also 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. Messages at the destination node are transferred to the PE through the ejection channel. Similar to the previous studies of [2, 15], this study considers the one-port router model where only one broadcast message can be injected into the network through different output channels.
The “Nearest-Corner-First” Broadcast Algorithm (2D mesh)

/* Input: source node \((S_x, S_y)\); M: Message; */
/* Output: All nodes receive a copy of M */

Let \(A^{x:y}\_\text{Corners}\) the fourth corners of the network

Let \(W\) be equal to \((N_x - 1)\) or \((N_y - 1)\)

Let \(A\) and \(B\) be the two dimensions of the nearest side (NS)

if \((S_x, S_y) \in A^{x:y}\_\text{Corners}\) then

\{Control field:= 11; Broadcast \((X,Y)\);\}

else

\{Choose the nearest corner:=Select \(C \in A^{x:y}\_\text{Corners}\); Control field:= 10; Send \(M\) to \(C\);\}

Procedure Broadcast \((X,Y)\)

\{Control field:= 11; \(A\)=select the first value of the \(C\) co-ordinate pair \((x, y)\); \(B\)=second value of the \(C\) co-ordinate pair \((x, y)\);

if \((A=0)\) then

while \((A < W)\) do

\{ if \((B=0)\) then Channels:=\(+\) else Channels:=\(-\);

\(A:=A+1; /* make a turn */ \}

else

while \((A > 0)\) do

\{ if \((B=0)\) then Channels:=\(+\) else Channels:=\(-\);

\(A:=A-1; /* make a turn */ \}

\}

end.

Fig. 1 A description of the new ‘NCF’ broadcast algorithm.

5. The Proposed “Nearest-Corner-First” Broadcast Algorithm

While most previous broadcast algorithm for the mesh have discussed in the context of deterministic routing, this section introduces the “Nearest-Corner-First” (or NCF for short) algorithm as a new broadcast algorithm that is based on adaptive routing. The NCF uses the Turn model discussed in [6] to achieve routing adaptivity while ensuring deadlock freedom (due to space limitation, we will omit the description of this routing algorithm. We refer the reader to [6] for more detail). While the Turn model prohibits just enough turns to ensure deadlock freedom, its adaptivity feature provides the NCF algorithm with a greater flexibility in choosing a network path for a message during a given message-passing step. Fig. 1 describes the broadcast operation in the NCF algorithm. The proposed algorithm exploits the features of the CPR to implement broadcast in two message-passing steps, thus considerably reducing the effects of both the network size and start-up latency. Examining Fig. 1 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. The NCF algorithm achieves this using the following rules:

i) If the source node \((S_x, S_y)\) is a corner node in the network, change the value of control field to “11” and go to step v,

ii) Find the nearest corner (or NC for short).

iii) Change the value of control field to “10” and pass the message across the shortest path towards the NC.

v) As a second message-passing step, the NC transmits the message to all the nodes of the network based on the turn model rules.

**Theorem 1:** The NCF algorithm in the mesh can be implemented in only one message-passing step if the source node is a corner node; otherwise, it can be implemented in two message-passing steps, irrespective of the network size.

**Proof:** Let the targeted system be a 2D mesh with \(N = N_x \times N_y\) nodes, and let the source node of the broadcast be at \((S_x, S_y)\). Without a loss of generality, let the node \((0, N_y - 1)\) be the source node as shown in Fig. 2a. Based on the rules of the NCF algorithm, the message translates across the first dimension (X in this case).

![Diagram of the message path in the nearest corner algorithm](image)

By turning 90-degree to the Y dimension, the message will be received by the all nodes of the X dimension that have the same value on the Y axis. However, in Fig. 2b an arbitrary source node has been chosen. In the first message-passing step, the message transmits to the nearest corner. As a second message-passing step, the message follows the rules of the Turn model [6] to be received by the all nodes of the network as is shown in Fig. 2b.

**Theorem 2:** The NCF algorithm is deadlock free in the all-port mesh.

**Proof:** To prove that the deadlock problem cannot occur during broadcast, we need to show that there is no cycle during message-passing step can occur [5]. Basically, two message-passing steps are required to implement a broadcast operation. In the first message-passing step, the source \((S_x, S_y)\) initiates the broadcast operation by transmitting the message towards the nearest corner across \(A^{i,j}\) nodes, where \(0 \leq i < S_x\) or \(S_x < j \leq N_y - 1\) and \(0 \leq j < S_y\) or \(S_y < j \leq N_y - 1\). In this step, the message selects one dimension to transmit, then it turns to the second dimension towards the nearest
corner, i.e., the message avoids any overlapping while the transmission. Clearly, the message follows the turn model rules in transmission in the second message-passing step.

6. Performance Comparison

In this section, we compare the performance of the proposed NCF algorithm to the well-known Recursive Doubling [2], Extended Dominating Node [15] and Network Partitioning algorithms [9]. In the rest of this section, we will use the short abbreviation NCF, RD, EDN and NP-D (following to refer to the four algorithms), respectively. Firstly, we will compare these broadcast algorithms in terms of the number of message-passing steps required. We then conduct a timing analysis to estimate the communication latency experienced by a broadcast message in the four algorithms. The RD was originally proposed by Barnett et al [2]. Since this algorithm cannot take any advantage of multiple port routers, it requires $\log_2 N$ steps to complete irrespective of the number of ports used in the network routers [15]. The EDN was proposed by Tsai and McKinley [15]. The authors in [15] have shown that the number of message-passing steps required in a network size of $(2^k \times 2^k)$ is $k + 1$. In [16], the NP-D has been proposed to achieve more parallelism during broadcasting operation. However, the NP-D does not minimise the message-passing steps required to perform broadcast operation in that it requires $k + 2$ to broadcasting in the $(2^k \times 2^k)$.

![Fig. 3: The Comparison of the message-passing steps](image)

Fig. 3 plots the number of message-passing steps required to perform a broadcast operation by the four algorithms in the $(2^k \times 2^k)$ mesh. For the sake of illustration, the network size was varied from 64 to 20376 nodes. The results reveal that while the number of steps required by the EDN, RD and NP-D algorithms increases with the system size, that required by the NCF is fixed regardless of the network size, i.e., it requires only two message-passing steps in all network sizes. The NCF achieves this minimal number of message-passing steps due to the use of the CPR approach. The rest of this section analyses the communication latency, that is the time required for a broadcast message to reach all the network nodes, in NCF, RD, EDN and NP-D. We use the resulting expressions for the communication latency to study the effects of important parameters, such as the message length, network size and start-up latency on the performance of the four algorithms.

The pipelining nature of wormhole switching makes latency less sensitive to message distance, especially when message are long. We define the communication latency for a broadcast operation as the time when a broadcast message is injected into the network until the last node in the network receives the message. In the absence of contention in the network, the communication latency, $\tau$, for a message length of $L$ flit can be generally estimated as

$$\tau_{\text{Broadcast}} = M\alpha + \beta D + \beta L + C\mu + \gamma$$

(1)

where $M$: is the number of copies of the broadcast message prepared by the source to be injected into the
network, $\alpha$ the sending latency for each message, $\beta$ the time required to transmit a flit on a channel, $D$: the distance between the source and destination of a message, $\gamma$: the receiving latency, $\mu$: the time required to change the header message and $C$: the number of message-passing steps required to deliver the message to all network nodes. Both the sending and receiving latency form the start-up latency, i.e. start-up latency = $\alpha + \gamma$ [2, 15]. Equation 1 will be used as a basis for analysing the execution time of the NCF, RD and EDN. Let us determine first the communication latency of the NCF. The communication latency, $\tau_{BCP}$, in the NCF, can be approximated by

$$
\tau_{NCF} = \begin{cases} 
\beta \min((N_x - 1 - S_x), S_x) + \min((N_y - 1 - S_y), S_y) \\
(N_x - 1 \times N_y - 1) + 2\beta L + 2\alpha + 3\mu + 2\gamma \\
\beta(N_x - 1 \times N_y - 1) + \beta L + \alpha + 2\mu + \gamma 
\end{cases} 
$$

(3)

The RD requires $\Log_2 N$ steps to complete a broadcast operation [2]. In a network size of $N=(2^k \times 2^k)$, the authors in [2] have shown that this algorithm requires $2k$ message-passing steps. The time required to send a broadcast message to all the network nodes can be written as [15]

$$
\tau_{RD} = (2k + m + 6)(\alpha + \beta L + D\beta + \gamma) 
$$

(4)

The EDN requires $k+1$ message-passing steps [15]. Based on the rules of this algorithm, a total time of $((3\alpha + \beta L + \gamma + d_1)\beta)$ is required for a message sent by the source to reach the four highest-levels of the network with $k-1$ message-passing steps. Two message passing steps are needed required by the source node to deliver the message to the four highest-level extended dominating nodes (see [15] for more detail). Therefore, the EDN requires a total time approximated by

$$
\tau_{EDN} = (k - 1)(3\alpha + \beta L + \gamma + d_1)\beta + 3\alpha + 2\beta L + 2\gamma + d_2\beta 
$$

(5)

where $d_1$ and $d_2$ represent the distances traversed by the message in the $k-1$ message-passing steps and the last two steps, respectively. Based on the rules of the NP-D algorithm, this algorithm requires $k+2$ message-passing steps on $(2^k \times 2^k)$ with total time required by mesh can be approximated by

$$
\tau_{NP-D} = ((d \log_2 2) + 2)\alpha + 2^{d+1}\beta + \left(\frac{5}{4} + \frac{1}{2^d}\right)\beta L + (k - d + 2)\alpha + 3(2^{k-2} - 2^d)\beta + \\
\frac{(k - d + 2)}{2^d} \beta L + ((d + 1)\log_2 2) + 2^{d-1}\alpha + 3(2^d - 1)\beta + 1 + \frac{((d + 1)\log_2 2) - 1}{2^d}\gamma 
$$

(6)

Table 2 demonstrates that the RD, EDN and NP-D suffer from the degrading effects of the start-up latency and network size. In contrast, our NCF algorithm has the lowest $\alpha$, $\gamma$ and $\beta$ factors. Moreover, the network size insensitivity is a unique feature of our broadcast algorithm. This is because in the NCF, 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. The NCF uses the CPR to fully exploit the insensitivity of the message distance inherent in wormhole switching to render the effects of the message distance on the overall communication latency negligible.

7. Conclusion
This study has proposed a new broadcast algorithm based on the Coded Path Routing (CPR) for one-port meshes. The proposed algorithm has the main advantage of requiring a fixed number of message-passing steps irrespective of the network size. Furthermore, our performance analysis has revealed that the proposed algorithm has superior performance characteristics than the existing Recursive Doubling, Extending
Dominating Nodes and Network Partitioning algorithms. The next step in our work is to extend the application of the CPR to devise new multicast algorithms and compare their performance with existing algorithms. Another possible line for future research is to extend of the CPR to support collective communication in other common multicomputer networks, such as tori and hypercubes.

Table 2. Comparison of broadcast latency in a \((2^k \times 2^k)\) mesh

<table>
<thead>
<tr>
<th>N=(2^k \times 2^k)</th>
<th>(\alpha)</th>
<th>(\gamma)</th>
<th>(\beta)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RD</td>
<td>2(k)</td>
<td>2(k)</td>
<td>2(k)</td>
</tr>
<tr>
<td>NCF</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>EDN</td>
<td>3(k)</td>
<td>3(k+1)</td>
<td>3(k+1)</td>
</tr>
<tr>
<td>NP-D</td>
<td>(\frac{(k-d+2^d+3+[(d+1)\log_2 2]+[d \log_2 2]}{4} + \frac{(k-d+2+[(d+1)\log_2 3])}{2^d})</td>
<td>3(x 2^k + 2^{d+1} - 3)</td>
<td></td>
</tr>
</tbody>
</table>

References