DISTRIBUTED RECOVERY BLOCK BASED FAULT-TOLERANT MULTICASTING

Gul N. Khan
Electrical and Computer Engineering
Ryerson University, 350 Victoria Street
Toronto, ON M5B 2K3 Canada
gnkhan@ee.ryerson.ca

G. S. Hura
Computer Science & Computer Engineering
University of Idaho, 1776 Science Center Drive
Idaho Falls, ID 83402 USA
ghura@if.uidaho.edu

Gu Wei
Advanced Ion Beam Technology Inc.
116 S Wolfe Road, Sunnyvale CA 94086 USA
guwei@yahoo.com

Abstract
This paper presents a fault-tolerant and adaptive multicasting method that employs a modified distributed recovery block (DRB) approach. The section of a distributed system spanning between the source and destination nodes is partitioned into series of overlapping DRB groups on the multicasting paths. Each DRB group consists of three nodes: a current node and two successor nodes. The source node becomes a current node in the first DRB group and it partitions the destination node-list to form the message header for delivering the message to its successor nodes. Each successor node receives the message, executes a recovery block try and performs the acceptance test. The successful successors become the current nodes of next DRB groups and update their destination lists. A prototype version of the method is simulated for 2D mesh and hypercube topologies to evaluate its performance.

Keywords: Adaptive routing; Distributed recovery block; Distributed Systems; Fault-tolerant multicasting.

1. INTRODUCTION

In parallel and distributed computer systems, message routing provides mechanism for communication between processing nodes. Efficient communication among nodes is critical to achieve higher system performance. The routing functions are often implemented in a distributed manner so that all routers work independently to accomplish the desired communications. Multicast and broadcast are collective communication operations where a message is delivered from a source to multiple destination nodes. Broadcast is a special case of multicast in which the addressed multicast group comprises all the nodes of a network. Multicast communication is directly used to implement data parallel programming operations such as replication and barrier synchronization and support of shared-data invalidation. It is also desirable to design communication strategies that multicast messages in the presence of faulty nodes or links of a distributed system. One major concern of multicasting in faulty networks is its routing ability for delivering messages to maximum number of destinations.

Multicast communication can be implemented either by multiple one-to-one routing or broadcasting. However, these alternative methods create unnecessary traffic and are likely to cause message delays, congestion and even deadlock. Wormhole routing based multicast communication is a complex process, as when a message header is blocked, the message flits remain stranded in the network. We are presenting a hierarchical tree-based multicasting method. Tree-based routing has also been used to support a restricted form of multicast in the nCUBE-2 [1]. Path-based routing techniques have also been proposed for multicasting. Path-based multicasting can be implemented by sorting destination in which they will be visited and placing the resulting list in the message header. The absence of deadlocks is guaranteed by eliminating cyclic dependencies between channels. Lin et al. have proposed adaptive multicast routing algorithms for 2D meshes [2].

Reachability, time steps required to route a message to desired destinations and the amount of traffic incurred in the system network are the main concerns of adaptive multicasting. Time steps is defined as the number of communication links a message has to travel to reach a
destination. Time optimality can be achieved for a destination if the length of the path from source to destination is the shortest under fault-free conditions. Reliable and efficient communication is key to the performance of parallel and distributed systems. It may be difficult to attain time optimality for a destination even if a shortest path exists in a faulty network. In certain cases, a message must be derouted in order to reach some of its destinations. In a multicasting scheme called *lookahead* [3], a fault-free look-ahead path is identified in case a message cannot reach some of the destinations due to failure. However, time optimality is not guaranteed by this scheme. Routing capability for a node is defined with respect to each distance. A number of algorithms that are based on the concept of routing capability have been reported in the past [4].

2. DRB BASED MULTICASTING

The fault-tolerant multicasting algorithm presented in this paper is implementable for various topologies having three or more communication links per node. We introduce the concept of distributed recovery block (DRB) [5] for fault-tolerant multicasts in faulty network. There are some assumptions that are made to present the multicasting technique in this paper:

- The source and destination nodes are fault-free.
- A node with no healthy links incident on it is considered faulty.
- Faults are non-malicious that means a failed component simply ceases to work.

The section of a parallel/distributed system spanning the source and destination nodes is dynamically partitioned into overlapping DRB groups. The formation of DRB groups is explained in Fig. 1 for multicasting a message from source node 11 to destination nodes (13, 22, 23, 32, 33, 42) in a 2D mesh. A DRB group has three nodes: the current node and two successor nodes. The current node delivers the message to its successor nodes. The successor nodes have a set of try and acceptance test and they receive the message from the current node and apply the acceptance test (AT). The time acceptance test is used to ensure a timely message delivery to both successors. In a fault-free situation, both successor nodes pass the AT, notify their success to each other and forward the message to next DRB group. In the case of a successor node failure, the successful successor takes over the routing for its faulty partner. If the successor nodes cannot communicate their failure, their failure is recognize after a time-out. If both successors fail, they will retry and if they are unable to recover after retry, some other successor nodes can be chosen for multicasting.

![Fig. 1. Formation of DRB groups](image)

We explain DRB based multicasting for a 2D mesh with two faulty links 20-21 and 11-12 as shown in Fig. 2. A message, $msg$ is to be multicast from source node 22 to destination node list, $DL = \{01, 02, 10, 11, 12, 20, 21\}$. The source node divides the $DL$ into two groups: $DL1 = \{12, 11, 01, 02\}$ and $DL2 = \{21, 20, 10\}$. It formats the original message in the form of $(\{12, 11, 01, 02\}, \{21, 20, 10\}, msg)$ and sends the message to its successors nodes 12 and 21 in one step-time. The partitioning of destination nodes depends on the location of source and its successor nodes in the network. The multicast routing decisions are made at each intermediate node. Upon receiving the message, each successor node first determines whether its address matches with the address of first destination address in the message header. If so, the address is removed form the header, and message is delivered to the node host. At this point, if the destination list is not empty, the message header is updated and message is forwarded toward the successor destination nodes. In this example, no fault occurs and successor node 12 removes itself from the destination list $DL1$ and form two new destination lists $DL1 = \{11, 01\}$ and $DL2 = \{02\}$. Then it sends the message $((11, 01), (02), msg)$ to its successor nodes 11 and 02. The link between node 11 and 12 is faulty and node 11 will not receive the message. On the other hand, successor node 02 receives the message successfully and updates its destinations according to the status of its partner. It will not receive the successful delivery information from its partner node 11. Successor node 02 removes itself from the destination list and forms a new destination list $DL2 = \{01, 11\}$.
and sends the message ([01,11], [ ], msg) to its successors 01 and 12. Similarly, on receiving the message, node 01 removes itself from the destination list and sends message ([11], [ ], msg) to node 11 and 00 respectively to complete the multicast.

![Fig. 2. A faulty 2D mesh](image)

Concurrently the second successor node, 21 of the source forms two destination lists [20] and [10] and sends the message ([20], [10], msg) to its successors 20 and 11. Successor node 20 doesn’t receive the message due to a faulty link and its partner successor 11 updates the destination lists accordingly and sends the message ([10], [20], [ ], msg) to its successors 10 and 01. This time no faults occur and the destination lists for nodes 10 and 01 will become [20] and [ ]. The message will reach all the destinations. The multicasting for the above example is also summarized in Fig. 3. It can be observed that the number of time steps is four for a 3x3 mesh and the message does not traverse to the destinations by the same link. The fault tolerant multicast algorithm is a tree-based multicast that achieves time optimality and reachability.

![Fig. 3. Multicasting in the faulty 2D mesh](image)

Now we present the multicasting method by assuming:

- src = source node;
- curr = current node;
- succ-1 and succ-2 = two successor nodes;
- partner = successor nodes are partner to each other;
- DL = destination list; AT = acceptance test

**current node: curr**

begin
  if (curr == src) then partition destination list DL;
  determine succ-1 and succ-2 from neighbor nodes;
  send (DL1, DL2, msg) to succ-1 and succ-2;
  /* where DL = DL1 + DL2 */
  receive the AT result from succ-1 and succ-2;
  exchange AT results between succ-1 and succ-2;
  if succ-1 or succ-2 fail then retry;
end.

**successor nodes: succ-x**

begin
  receive (DL1, DL2, msg) from curr;
  execute the AT;
  send AT result to curr;
  receive partner’s AT result from curr;
  if (succ_x ∈ DLx) && (succ_x passes the AT)
    /* Am I a destination node? */
    then accept msg locally;
  else if (succ_x ∈ DLx) && (succ_x fails the AT)
    then { retry }
  if (succ_x passes the AT) then {
    DL = update_destination_list (succ_x);
    if (DL ≠ ∅) then curr = succ_x;
  }
end.

**update_destination_list (succ_x)**

begin
  if (succ_x ∈ DLx) then { DLx = DLx − succ_x }
  if (succ-x pass the AT) then DL = DL − DLy;
  /*remove the partner’s list*/
  divide the DL into new sub-groups DL1 and DL2;
  format the message as (DL1, DL2, msg)
end.

### 3. EXPERIMENTAL RESULTS

A prototype of the DRB-based fault-tolerant multicasting is implemented by using an IBM SP parallel system. A number of simulations are performed by multicasting message in injured 2D mesh and hypercube. The nodes of an IBM SP parallel system are interconnected by a switch for direct communication. However, for this simulation, the required topologies are configured by restricting the nodes to communicate only with its topological neighbours directly. We compare the message latency of different faulty scenarios for a message size of 256 bytes for 2D mesh and hypercube topologies. Consider the faulty mesh of Fig. 2 and a...
message is to be broadcast from source node 22. The overall multicasting process has been presented earlier in this paper. Message latency is presented in Fig. 4 by varying the number of destinations. It is observed that the latency remained unchanged for few destinations. However, when the number of destination nodes increase, the latency increases by 16% as compared to fault-free situation. It happens due to link failure as it takes more hops for the message to make a detour to its destinations. However, the time steps and network traffic remain almost constant. Time optimality may not be guaranteed for some destinations even if a shortest path exists between the source and destination nodes.

**Fig. 4. Multicasting latency in 2D mesh**

Similar simulations are conducted for an injured hypercube of degree three as given in Fig. 5. The time step for the broadcast is four while the total number of distinct links the message traverses for reaching all the destinations is thirteen. Message latencies are measured for fault-free and link failure cases. The results are presented in Fig. 6 that are similar to the simulation result for mesh topology discussed earlier.

**Fig. 5. An injured hypercube**

### 4. CONCLUDING REMARKS

Fault-tolerant multicasting is an important and new application of distributed recovery block approach. A prototype of the DRB based fault-tolerant and adaptive multicasting is implemented for 2D mesh and hypercube topologies. The method can be extended to other topologies with a node connectivity of more than three. Multicasting technique has been investigated and analyzed in detail for various types of faults and for different number of destinations. Investigation results show that the DRB based multicasting is an effective way of assuring the message delivery to all the intended destinations. The proposed routing technique does not require the nodes to keep any fault information. It is also not necessary for the nodes to know the status of its neighboring nodes.

**Fig. 6. Message latency in faulty hypercube**

### Acknowledgements

Part of this research by the first author is supported by a grant from NSERC Canada.

### References


