Supporting Fully Adaptive Routing in InfiniBand Networks

J.C. Martínez, J. Flich, A. Robles, P. López and J. Duato

I. INTRODUCTION

INFINIBAND [11] is a new interconnect standard for communication between processing nodes and I/O devices, as well as for interprocessor communication (IPC). The InfiniBand Architecture (IBA) [12] is designed around a switch-based interconnect technology with high-speed serial point-to-point links. IBA supports any topology defined by the user in order to provide wiring flexibility and incremental expansion capability.

More than 180 companies, including the leading computer manufacturers, support the InfiniBand initiative. It was designed to solve the lack of high bandwidth, reliability, availability and scalability of existing server I/O technologies. However, the spectrum of possible application domains for InfiniBand is wider, including I/O interconnect, system area networks (SAN), storage area networks (STAN), and local area networks (LAN).

InfiniBand could be used as a SAN fabric for high performance server clusters or commodity clusters formed by PCs or workstations. In particular, InfiniBand can provide the high bandwidth and low end-to-end latency required for enhancing commodity clusters that can be used as a cost-effective alternative to parallel computers.

Routing in IBA subnets is distributed and deterministic, based on forwarding tables stored in each switch and which only consider the packet destination ID for routing packets [12]. The routing tables only store one output link per destination. Deterministic routing is used in many network interconnects due to its simplicity [1], [14]. However, deterministic routing algorithms do not make effective use of network links because a unique path is provided for each source-destination pair.

IBA allows the use of alternative paths between any source-destination pair 1. The final path can be selected at each source node according to certain criterion (random, round-robin, etc.). However, by using alternative paths selected at the source node, the network performance is hardly improved [5], [6].

On the other hand, adaptive routing [3] dynamically builds the path used by a packet, selecting the channels along the path considering network status information. With adaptive routing, each switch selects the output port to forward a packet from a set of routing options, avoiding congested areas and increasing network performance. Several adaptive routing algorithms have been proposed in the literature [2], [16], [10], showing that network performance can be strongly increased.

However, adaptive routing has some drawbacks. The first one is that it increases switch complexity. The second one is the fact that it cannot guarantee in-order packet delivery. However, in most MPI-based parallel applications, there exists a certain percentage of traffic that may be delivered out-of-order. Moreover, in-order packets could also use adaptive routing if packets were reordered at the destination host before being delivered.

II. MOTIVATION

At first glance, InfiniBand specifications [12] do not support distributed adaptive routing. Effectively, IBA specs state that forwarding tables must provide only one physical output port per destination node. However, IBA specs do not define the internal architecture of a switch. Indeed, IBA switches already support distributed routing. Thus, enhancing switch capabilities to support adaptive routing could be feasible.

In [13], we propose a simple strategy to allow adaptive routing in IBA while still maintaining compatibility with the IBA specs. Indeed, this proposal can be considered as an extension of IBA specs, adding more functionality to them. In this paper, we provide support to fully adaptive routing algorithms [4] in IBA switches. In particular, we implement an extension to virtual cut-through of the MA routing algorithm [16] for networks with irregular

1 By means of the virtual addressing scheme of IBA [12].
topology. The mechanism proposed in this paper does not need to add resources to the IBA switches. Only, the existing resources must be appropriately arranged.

The rest of the paper is organized as follows. Section III describes the proposed mechanism. In Section IV some performance evaluation results are shown. Finally, in Section V some conclusions are drawn.

III. SUPPORTING ADAPTIVE ROUTING IN IBA

This section contains a short description of the proposed mechanism. A more detailed description can be found at [13].

An IBA network is composed by end nodes interconnected by switches. Each end node contains one or more channel adapters (CA). Each channel adapter contains one or more ports. Each port has a local identifier (LID) assigned by the local subnet manager, which is unique within the subnet. IBA switches route packets based on forwarding tables stored at each switch, addressed by the packet destination local identifier (which is referred to as the destination LID or DLID). As this table only returns one output port to forward the packet towards its destination, IBA switches do not support adaptive routing.

A. Providing Multiple Routing Options

In order to support adaptivity, each switch should supply several feasible output ports when a packet is routed. We can use a trick to provide multiple routing options in IBA switches. IBA allows a single destination to be assigned not only a unique address but a range of them by defining a LID Mask Control or LMC [12]. The LMC specifies the number of least significant bits of the LID that a physical port masks (ignores) when it validates that a packet DLID matches its assigned LID. As these bits are not ignored by the switches from the subnet point of view, each CA port has been assigned up to 2LMC consecutive addresses. Each CA port will accept all packets destined for any valid address within its range. Notice that IBA specifications allow for a maximum of 128 different addresses per destination port. Thus, a maximum of 128 different routing options could be provided with the proposed scheme, which seems more than required.

The idea is to assign to each port the same number of addresses as the number of routing options at each switch. In this case, there will be a range of consecutive addresses assigned to the same host. When the switch has to select the output port for any packet, it selects all the output ports that are assigned to the packet destination host, despite selecting only one output port. In this way, the forwarding table will provide more than one output port for each packet.

This mechanism allows source hosts to select deterministic or adaptive routing, on a per packet basis. When a source host selects the least virtual address (d) of the destination host, only an output port will be selected at each switch. This output port will be always the same for each destination host in order to provide deterministic routing (in-order delivering). When a source host selects the address d + 1, all the output ports will be provided in order to provide adaptive routing.

In order to access several entries in the forwarding table, they can be accessed sequentially or a multi-port memory can be used. However, if a linear forwarding table2 [12] is used at switches, a simple implementation can be done by organizing the forwarding table as an interleaved memory composed by several modules that are selected by the least significant bits.

To sum up, Figure 1 shows the implementation of the mechanism when two routing options are provided at each switch. The destination field of the packet (the DLID) is used to access the forwarding table, obtaining simultaneously two output ports. To allow these two simultaneous accesses, the forwarding table is organized as two interleaved modules. In order to select the switch output port that will be finally used, the least significant bit of the DLID is first checked. If it is set to zero, deterministic routing is required for the packet, so the output port that corresponds to the first address is selected. Otherwise, two routing options are selected. The final selection can be done either immediately or at the internal switch arbitration time, and either considering the status of the output ports or performing a static selection.

B. Support for Adaptive Routing with Escape Paths

The mechanism allows the use of adaptive routing algorithms with escape paths. Therefore, the fully adaptive routing algorithm [4] can be used in IBA.

We could split each IBA VL into two queues, the adaptive and the escape queues. Both queues are multiplexed onto the corresponding VL. However, we have not support to identify from which queue data is being transferred or to manage the buffer space available at each queue. To solve this problem, we propose (see Figure 2) to divide the physical buffer assigned to each VL into two logical queues that will implement the adaptive and the escape queue, respectively. The first part of the physical buffer corresponds to the adaptive queue, whereas the second part of it is the escape queue. Notice that the entire VL is treated as a unique queue. Therefore, the escape queue will be used only when the adaptive queue is full.

However, the packets stored in the escape queue must be able to be routed and forwarded independently of the ones stored in the adaptive one. This can be accomplished in the proposed organization by arranging two connections from each VL to the internal switch, one located at the head of the adaptive queue and another located at the head of the escape queue. In Figure 2, a multiplexer is used in order to select a packet either from the adaptive or the escape queue. Since escape and adaptive queues are parts of the same queue, a packet may be transferred from the escape queue to the adaptive queue. As can be seen in [4], this does not lead to deadlock.

Once the forwarding table offers the adaptive and escape routing options (leading to output ports op0 and

2The linear forwarding table provides a simple map from LID to output port. In other words, the table contains only output ports and the LID acts as an index into the table.
The escape routing option can be used at any time. Notice that packets forwarded through this routing option will be stored either in the adaptive or the escape queues depending on the number of available credits (Figure 2), and properly select the output port to use. The switch can either decide to forward the packet through the escape queue (once there are enough available credits) or to take the decision later.

As the buffer associated to VLs is now divided into two logical queues (adaptive and escape) and virtual cut-through is used, each one of them should be able to store an entire packet. This can be accomplished either by increasing buffer size accordingly or by reducing the Maximum Transfer Unit (MTU).

As switches may extract a packet from the head of the escape queue, a packet may overtake another packet that has been placed at the adaptive queue in the same VL. This may produce out-of-order delivering even when using always the same routing option per destination. Switches may avoid this problem using a pointer to the first “deterministic” packet in the VL. The packet pointed by this pointer is considered as the head of the escape queue. Therefore, “deterministic” packets cannot bypass other “deterministic” ones.

**IV. PERFORMANCE EVALUATION**

In this section we will evaluate the impact on network performance of the proposed strategy. For this purpose, we have developed a detailed simulator that allows us to model the network at the register transfer level following the IBA specifications [12]. First, we will describe the IBA subnet model defined in the specs together with the simulator parameters and the modeling considerations we have used in all the evaluations. Then, we will evaluate the adaptive technique proposed under different topologies and different traffic patterns.

**A. The IBA Subnet Model**

The IBA specification defines a switch-based network with point-to-point links, allowing the user to define any topology. The network allows the communication between end-nodes. The end-nodes are attached to switches using the same kind of links used between switches.

Packets are routed at each switch by accessing the forwarding table, that contains the output port to be used at the switch for each possible destination. Several routing options are provided based on the strategy proposed in this paper. In particular, the routing options will be computed by using the FA routing algorithm proposed in [4], which uses the up*/down* routing for computing the escape paths. The output port is selected at arbitration time considering the status of the requested output ports and the number of credits available.

Switches can support up to 16 virtual lanes. VLs can be used to form separate virtual networks. We will use a non-multiplexed crossbar on each switch. This crossbar supplies separate ports for each VL. Buffers will be used both at the input and the output side of the crossbar. Buffer size will be fixed in both cases to 1KB. Hence, the adaptive and escape queues implemented on each buffer will be 512 bytes depth.

The switch routing time will be set to 100ns, including the time to access the forwarding tables, the crossbar arbiter time, and the time to set up the crossbar connections.

Links in InfiniBand are serial. 10/8 coding [12] is used. In the simulator, the link rate will be fixed to the 1X and 4X configuration [12] (2.5 and 10 Gbps). We will model 20m copper cables with a propagation delay of 5ns/m.

The IBA specification defines a credit-based flow control scheme for each virtual lane with independent buffer resources. A packet will be transmitted over the link if there is enough buffer space (measured in credits of 64 bytes) to store the entire packet. IBA allows the definition of different MTU values for packets, ranging from 256 to 4096 bytes. We use a MTU of 256 bytes. Additionally, the virtual cut-through switching technique is used.

Several packet destination distributions will be used: uniform, bit-reversal, and hot-spot. In the latter case, a node is randomly selected and a percentage (we used 5%, 10%, and 20%) of traffic is sent to this host. 32 and 256-byte packets will be used. In all the presented results, we will plot the average packet latency measured in nanoseconds versus the average accepted latency in Figure 2)
B. Evaluation Results

In this section we analyze the influence on network performance when using IBA switches with adaptive routing capabilities. First, we analyze the influence of the percentage of adaptive traffic on network performance. Then, we analyze how the network connectivity affects network performance.

B.1 Influence of the Percentage of Adaptive Traffic

Figures 3.a, 3.b, 3.c, and 3.d show the simulation results for the FA routing when varying the percentage of adaptive traffic from 0% (deterministic traffic) up to 100% for network sizes of 8, 16, 32, and 64 switches, respectively. In this case, forwarding tables provide two routing options at most and four links (4X each one) are used in each switch to connect to other switches. Uniform packet destination distribution and 32-byte packets are used.

As it can be seen, the improvement on performance achieved by using IBA switches with support for adaptive routing linearly increases with the percentage of applied adaptive traffic. However, for the 8-switch network, when 75% or 100% of adaptive traffic is injected, the FA routing algorithm almost obtains the same network throughput. On the contrary, for the 64-switch network, the difference in network throughput when injecting 75% and 100% of adaptive traffic is greater.

Table I shows minimum, maximum, and average factors of throughput increase for different network sizes and different packet sizes when using 100% adaptive traffic and 4X links. As we can see, network throughput benefits increase as network size increases. In particular, a uniform traffic pattern with 32-byte packets, when using networks with two routing options and 4 links connecting switches, network throughput is increased, on average, from 1.3 to 3.28, depending on network size.

The higher throughput increase observed for large networks with respect to small networks is due to the fact that the up*/down* routing does not scale well. Therefore, with 0% of adaptive traffic, as network size increases, the up*/down* routing tends to use longer non-minimal paths and also to unbalance the traffic, congesting the switches near the root switch [7]. Hence, packets benefit more from using adaptive routing.

Table I shows results for other traffic patterns. The higher the percentage of hot-spot traffic, the lower the factors of throughput improvement. This is because traffic around the hot-spot tends to concentrate, spreading the congestion through the network. Better results are obtained for other traffic patterns (See Table I).

Indeed, for the bit-reversal traffic pattern (which creates some local congestion areas), similar results to the uniform traffic pattern are obtained, as shown in Table I. We can observe that the use of adaptive routing causes throughput to increase, on average, by a factor of 1.58 for 8-switch networks and 2.83 for 64-switch networks.

Finally, we can observe that qualitatively similar results were obtained for long packets.

B.2 Influence of Increasing the Number of Routing Options and Network Connectivity

With a connectivity of 4 links per switch, it is not worth providing more than two routing options per switch (See Table III). For instance, only 17.48% of the destinations provide more than two routing options in a 64-switch network. However, when using 6 links to connect to other switches we can observe that this percentage is increased.

Table I (right side) shows the throughput improvement results when switches have 6 ports available to connect to other switches and forwarding tables provide up to four routing options for uniform traffic. With 4 links connecting switches and up to three routing options per destination at each switch, throughput is slightly increased (3.28 vs 3.50 for 64-switch networks and 32-
Fig. 3. Average packet latency vs. traffic. 1 Virtual lane and 4 links connecting switches. 4X links. Uniform traffic pattern. Network size is (a) 8, (b) 16, (c) 32, and (d) 64 switches. Packet size is 32 bytes.

TABLE I

<table>
<thead>
<tr>
<th>Sw</th>
<th>MRLS</th>
<th>Traffic</th>
<th>32-byte packets</th>
<th>256-byte packets</th>
<th>Sw</th>
<th>MRLS</th>
<th>Traffic</th>
<th>32-byte packets</th>
<th>256-byte packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>2/4</td>
<td>Unif.</td>
<td>1.17/1.42/1.90</td>
<td>1.27/1.52/1.38</td>
<td>8</td>
<td>3/4</td>
<td>Unif.</td>
<td>1.13/1.20/1.92</td>
<td>1.27/1.40/1.18</td>
</tr>
<tr>
<td>16</td>
<td>2/4</td>
<td>Unif.</td>
<td>1.14/2.03/1.71</td>
<td>1.25/1.62/1.45</td>
<td>16</td>
<td>3/4</td>
<td>Unif.</td>
<td>1.23/2.04/1.71</td>
<td>1.25/1.82/1.52</td>
</tr>
<tr>
<td>32</td>
<td>2/4</td>
<td>Unif.</td>
<td>1.77/1.46/2.38</td>
<td>1.44/2.32/1.86</td>
<td>32</td>
<td>3/4</td>
<td>Unif.</td>
<td>1.77/2.26/2.45</td>
<td>1.56/2.51/1.99</td>
</tr>
<tr>
<td>64</td>
<td>2/4</td>
<td>Unif.</td>
<td>2.82/4.23/2.28</td>
<td>2.09/2.60/2.39</td>
<td>64</td>
<td>3/4</td>
<td>Unif.</td>
<td>2.97/4.70/3.50</td>
<td>2.09/2.79/2.52</td>
</tr>
<tr>
<td>8</td>
<td>2/3</td>
<td>HS 5%</td>
<td>1.23/1.36/1.40</td>
<td>1.27/1.69/1.42</td>
<td>8</td>
<td>3/4</td>
<td>Unif.</td>
<td>1.18/1.68/1.45</td>
<td>1.36/2.08/1.69</td>
</tr>
<tr>
<td>16</td>
<td>2/3</td>
<td>HS 5%</td>
<td>1.25/2.01/1.61</td>
<td>1.26/1.77/1.47</td>
<td>16</td>
<td>2/6</td>
<td>Unif.</td>
<td>1.51/2.43/2.02</td>
<td>1.69/2.40/2.00</td>
</tr>
<tr>
<td>32</td>
<td>2/3</td>
<td>HS 5%</td>
<td>1.50/2.11/1.87</td>
<td>1.47/2.08/1.76</td>
<td>32</td>
<td>2/6</td>
<td>Unif.</td>
<td>2.49/3.11/2.88</td>
<td>2.36/2.91/2.67</td>
</tr>
<tr>
<td>64</td>
<td>2/3</td>
<td>HS 5%</td>
<td>1.70/2.28/1.97</td>
<td>1.71/2.39/1.98</td>
<td>64</td>
<td>2/6</td>
<td>Unif.</td>
<td>2.59/3.47/3.07</td>
<td>2.54/3.20/2.93</td>
</tr>
<tr>
<td>8</td>
<td>2/4</td>
<td>HS 10%</td>
<td>1.09/1.65/1.28</td>
<td>1.09/1.66/1.26</td>
<td>8</td>
<td>3/6</td>
<td>Unif.</td>
<td>1.20/1.74/1.49</td>
<td>1.37/2.31/1.82</td>
</tr>
<tr>
<td>16</td>
<td>2/4</td>
<td>HS 10%</td>
<td>1.12/1.43/1.30</td>
<td>1.22/1.48/1.33</td>
<td>16</td>
<td>3/6</td>
<td>Unif.</td>
<td>1.56/2.43/2.01</td>
<td>1.75/2.62/2.16</td>
</tr>
<tr>
<td>32</td>
<td>2/4</td>
<td>HS 10%</td>
<td>1.14/1.57/1.38</td>
<td>1.15/1.62/1.40</td>
<td>32</td>
<td>3/6</td>
<td>Unif.</td>
<td>1.56/2.47/2.01</td>
<td>1.75/2.62/2.16</td>
</tr>
<tr>
<td>64</td>
<td>2/4</td>
<td>HS 10%</td>
<td>1.27/1.71/1.35</td>
<td>1.24/1.61/1.36</td>
<td>64</td>
<td>3/6</td>
<td>Unif.</td>
<td>2.56/3.47/3.07</td>
<td>2.54/3.20/2.93</td>
</tr>
<tr>
<td>8</td>
<td>2/3</td>
<td>BR</td>
<td>1.06/1.69/1.37</td>
<td>1.14/1.82/1.39</td>
<td>8</td>
<td>4/6</td>
<td>Unif.</td>
<td>1.20/1.74/1.49</td>
<td>1.37/2.31/1.82</td>
</tr>
<tr>
<td>16</td>
<td>2/3</td>
<td>BR</td>
<td>1.25/1.91/1.56</td>
<td>1.22/1.61/1.39</td>
<td>16</td>
<td>4/6</td>
<td>Unif.</td>
<td>1.56/2.57/2.05</td>
<td>1.75/1.67/2.21</td>
</tr>
<tr>
<td>32</td>
<td>2/4</td>
<td>BR</td>
<td>1.51/2.95/1.33</td>
<td>1.48/2.20/1.85</td>
<td>32</td>
<td>4/6</td>
<td>Unif.</td>
<td>2.88/2.48/3.17</td>
<td>2.70/2.83/3.02</td>
</tr>
</tbody>
</table>

V. CONCLUSIONS

We have proposed a simple mechanism to enhance IBA switch capabilities to support adaptive routing while maintaining compatibility with IBA specs. For this aim, forwarding tables are arranged in such a way that they can provide several routing options at the same time. Also, the logic circuitry necessary to support fully adaptive routing algorithms (adaptive and escape queues and the proper utilization of credits to avoid deadlock) has been proposed. Also, adaptive routing can be enabled or disabled on a per-packet basis by the running application.

The proposed mechanism has been evaluated using the fully adaptive routing scheme proposed in [4]. Results show that enhancing IBA switches with adaptive routing noticeably increases network performance. This is specially significant for large networks. As network connectivity increases, higher throughput improvement is obtained. In particular, network can be improved up to a factor of 3.9.
Although the proposed mechanism consumes a virtual address per routing option, the number of required addresses remains low and it is not a scarce resource. Also, evaluation results show that by using only two routing options per destination port at each switch, roughly 90% of the maximum throughput improvement is achieved.

We recently proposed some effective strategies to improve IBA network performance [8, 9] by allowing most packets to be routed through minimal paths and providing better traffic balance. These strategies make an efficient use of virtual lanes that are not used by QoS purposes. As future work we plan to combine this mechanism with these strategies in order to boost network performance further.

### References


