Supporting Fully Adaptive Routing in InfiniBand Networks *

J.C. Martínez, J. Flich, A. Robles, P. López and J. Duato
Dept. of Computer Engineering (DISCA)
Universidad Politécnica de Valencia
E-mail: jc@disca.upv.es

Abstract

InfiniBand is a new standard for communication between processing nodes and I/O devices as well as for interprocessor communication. The InfiniBand Architecture (IBA) supports distributed routing. However, routing in IBA is deterministic because forwarding tables store a single output port per destination ID. This prevents packets from using alternative paths when the requested output port is busy. Despite the fact that alternative paths could be selected at the source node to reach the same destination node, this is not effective enough to improve network performance. However, using adaptive routing could help to circumvent the congested areas in the network, leading to an increment in performance.

In this paper, we propose a simple strategy to implement forwarding tables for IBA switches that support adaptive routing while still maintaining compatibility with the IBA specs. Adaptive routing can be enabled or disabled individually for each packet at the source node. Also, the proposed strategy enables the use in IBA of fully adaptive routing algorithms without using additional network resources to improve network performance. Evaluation results show that extending IBA switch capabilities with fully adaptive routing noticeably increases network performance. In particular, network throughput increases up to an average factor of 3.9.

Keywords SANs, InfiniBand, adaptive routing, virtual addressing.

1 Introduction

InfiniBand [14] is a new interconnect standard for communication between processing nodes and I/O devices, as well as for interprocessor communication (IPC). The

---

*This work was supported by the Spanish Ministry of Science and Technology under Grants TIC2000–1151–C07 and 1FD97-2129, by Generalitat Valenciana under Grant GV00-131-14, and by JJ.CC. de Castilla-La Mancha under Grant PBC-02-008.

InfiniBand Architecture (IBA) [15] is designed around a switch-based interconnect technology with high-speed serial point-to-point links IBA supports any topology defined by the user, including irregular topologies, 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). In most of these applications areas, InfiniBand competes with other well-accepted industry standards, as PCI-based technologies [19] in the I/O interconnect area, or Fibre-Channel [6] and Ethernet [22] in the LAN and STAN areas.

On the other hand, in SAN environments, while several interconnects like Myrinet [18], ServerNet II [12] or Fibre Channel can be applied, nowadays there is no widely accepted technology. In particular, InfiniBand could be used as a SAN fabric for high performance server clusters or commodity clusters formed by PCs or workstations. At the server cluster level, InfiniBand has the advantage of handling both interprocess communication and shared storage I/O on the same network fabric. In turn, InfiniBand can provide the high bandwidth and low end-to-end latency required for enhancing commodity clusters. In particular, commodity clusters could be used as a cost-effective alternative to parallel computers, which are commonly based on expensive proprietary systems.

The increment in the number and power of cluster nodes is likely to cause more and more throughput requirements to the interconnection network. Despite the fact that IBA switches provide high channel bandwidth, it is likely that the fabric interconnect will again become the bottleneck in the future, specially when using switches with a low number of ports and low-end links (1X links at 2.5 Gbps). In this way, improving the throughput levels achieved by the
switch fabric in IBA will become a key issue. This could be achieved by using efficient routing schemes.

Routing in IBA subnets is distributed, based on forwarding tables stored in each switch and which only consider the packet destination ID for routing packets [15]. IBA routing is deterministic since the routing tables only store one output link per destination ID. Moreover, IBA switches support virtual lanes, but they are mainly intended for providing QoS to applications. Moreover, virtual lanes cannot be dynamically selected at each switch.

As shown in [4], the applied routing scheme has a great influence on network performance. Deterministic routing is used in many network interconnects due to its simplicity [1, 20]. However, deterministic routing algorithms do not make effective use of network links because a unique path is provided for each source-destination pair. If one or more channels of this path are busy, then the traffic between that source-destination pair will be delayed. This could be avoided by dynamically selecting an alternative path.

IBA allows the use of alternative paths between any source-destination pair. 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 overall network performance is hardly improved [7, 8].

On the other hand, adaptive routing [4] dynamically builds the path used by a packet, selecting the channels along it considering network status information. With adaptive routing, each switch selects the output port to forward a packet from a set of routing options. This selection can be done upon the status of the channels (busy or free) and therefore, the busy channels are skipped. Hence, adaptive routing allows packets to circumvent the congested areas, thus making better use of network resources and increasing network performance. Several adaptive routing algorithms have been proposed in the literature [2, 24, 13], showing that network performance can be strongly increased. Some of these proposals have been later applied to commercial systems, such as the Cray T3E [21] and the Compaq Alpha 21364 [17].

Adaptive routing algorithms may provide either a few alternative paths between some pairs of nodes or the full set of possible paths for every pair of nodes. In the former case, the routing algorithm is known as partly adaptive, like up*/down* [20], whereas in the latter case the algorithm is fully adaptive, like Minimal Adaptive (MA) [24].

However, adaptive routing has some drawbacks. The first one is that it increases switch complexity. In particular, distributed routing (as opposed to source routing) [4] and some channel selection logic are required. 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 the traffic that may be delivered out-of-order. In these cases, it could be worthwhile allowing packets to be adaptively routed to improve network performance. Moreover, in-order packets could also use adaptive routing if packets were reordered at the destination host before being delivered.

2 Motivation

At first glance, InfiniBand specifications [15] 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 this paper, we take on such a challenge. 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. This could be specially interesting for IBA switch manufacturers in order to improve the capabilities of future IBA products.

We are interested in providing support to fully adaptive routing algorithms in IBA switches. In particular, we will implement an extension to virtual cut-through of the MA routing algorithm for networks with irregular topology [24]. The mechanism proposed in this paper to support adaptive routing does not need to add resources to the IBA switches. Only, the existing resources must be appropriately arranged.

The new switch capabilities for supporting adaptive routing could be used by MPI-based parallel applications, if required, to improve their performance. This is the case when the applications have enough parallelism and they are able to initiate many concurrent non-blocking message transmissions. In particular, parallel applications could benefit more from using fully adaptive routing in networks with irregular topologies, which can be used in cluster interconnects. This is because packets have more difficulties to be routed through minimal paths in irregular networks. However, applying fully adaptive routing would allow most of the packets to follow minimal paths. Thus, improving the overall network performance.

The rest of the paper is organized as follows. Section 3 describes the implemented fully adaptive routing algorithm. In Section 4 the proposed mechanism is described, showing some performance evaluation results in Section 5. Finally, in Section 6 some conclusions are drawn.

---

1By means of the virtual addressing scheme of IBA [15].
3 Fully Adaptive Routing Algorithm for Networks with Irregular Topology

A general methodology for the design of adaptive routing algorithms for wormhole-switched networks with irregular topology was proposed in [23, 24]. The aim of this methodology is to provide minimal routing between every pair of nodes, as well as to increase adaptivity. Recently, this methodology was particularized for virtual cut-through (VCT) networks [5], as it is the case of InfiniBand. In summary, this methodology starts from a deadlock-free routing algorithm for a given interconnection network and associates two queues with each physical link, referred to as adaptive and escape queues, respectively. Each queue can store one or more packets. Then, the routing function is extended so that adaptive queues are used for fully adaptive minimal routing. Escape queues are used in the same way as in the original routing function. When a packet is routed at a switch, it first tries to reserve an adaptive queue. If all of the suitable outgoing adaptive queues are busy, then an escape queue is selected.

Unlike the original methodology [23, 24], a packet is allowed to reserve again an adaptive queue after using an escape queue. This increased routing freedom is possible in VCT because packets are fully stored in buffers at intermediate switches when they block, thus releasing the network resources occupied at previous switches.

The routing algorithms designed with this methodology are deadlock-free, provided that the original routing algorithm [5] is deadlock-free. Indeed, this methodology can be applied to any deadlock-free routing algorithm. In this paper we will apply it to the well known up*/down* [20] routing algorithm. The resulting routing algorithm will be referred to as Fully Adaptive (FA).

It can be easily proved that the resulting routing algorithm is deadlock-free by using the design methodology proposed in [3]. Moreover, taking into account that the resulting routing scheme allows packets to move from adaptive to escape queues and vice versa, and that routing on escape queues may be non-minimal, livelock may arise. We apply a similar strategy as the one used in [16] giving preference to minimal paths over non-minimal ones. This guarantees that as time goes to infinity, the probability of livelock goes to zero.

4 Supporting Adaptive Routing in IBA

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. In order to support adaptivity, each switch should supply several feasible output ports when a packet is routed.

4.1 Providing Multiple Routing Options

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 [15]. 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 $2^{LMC}$ consecutive addresses. Each CA port will accept all packets destined for any valid address within its range.

The idea is to assign to each destination port the same number of addresses as the number of desired routing options at each switch. Assume that we want two routing options per switch. This consumes two addresses per destination port (for instance, the addresses $d$ and $d+1$ refer to the same destination port $D$). As these addresses are different from the subnet point of view, all network switches will store the two routing options in the forwarding tables, in the positions assigned to $d$ and $d+1$.

When a packet destined to destination port $D$ (with virtual addresses $d$ or $d+1$) arrives at a switch, it will access not only to one address in the forwarding tables but also the other one ($d$ and $d+1$), thus providing two routing options to the packet. In order to access the two addresses in the forwarding table, either both of them are accessed sequentially or a multi-port memory is used. Both options slow down routing, as the former requires two table accesses and the latter may increase table access time. However, if a linear forwarding table \[^2\] [15] is used at switches, a simple implementation can be done by organizing the forwarding table as an interleaved memory composed by two modules that are selected by the least significant bit. The addressing logic is designed in such a way that it returns simultaneously the data found at addresses $d$ and $d+1$.

Although the two tables are internally accessed in parallel for routing, the interleaved organization makes the table to be externally viewed as a single table. Therefore, we conform with IBA specifications. Forwarding tables are

\[^2\] The 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.
filled by the subnet manager [15] at initialization time. It assigns a switch output port to each table address. Hence, to fully support the proposal we made in this paper, once the different routing choices have been computed for a given destination port, the subnet manager stores them in a range of addresses of the forwarding tables, as if they were different destinations.

In order to provide more than two routing options (say \( x \)), it is required to reserve a range of \( x \) addresses for each port and fill-up this range in the forwarding tables with the correct switch output ports. Most importantly, the forwarding table must be organized in such a way that it returns the \( x \) output ports each time it is accessed while maintaining a linear interface to the subnet manager. If \( x \) is a power of two, this can be accomplished again by arranging the forwarding table as an interleaved memory.

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. However, some of these addresses may be required for other purposes. In particular, some addresses may be required to provide fault-tolerant paths by the Automatic Path Migration (APM) method defined in the specs [15]. However, the entire set of paths can be divided (by using separate bits in the LMC) to allow the coexistence of both mechanisms (adaptivity and multiple paths)\(^3\). In practice, as there are a little number of different paths to forward a packet to its destination from a given switch, with a low number of routing options is enough (see section 5 for more details). Hence, the proposed mechanism consumes some virtual addresses but i) the number of required addresses is low and ii) it is not a scarce resource.

### 4.2 Enabling/Disabling Adaptive Routing

As we stated in Section 2, adaptive routing may not be useful every time. In fact, some data streams will have to be sent by using deterministic routing in order to guarantee inorder delivery and low latency. In order to deal with these situations, we have provided a way of enabling/disabling the proposal described in this paper. Sender nodes are responsible of selecting an adaptive or deterministic path for a given packet before injecting it into the network. This is achieved by using different destination addresses in the packet header. In fact, we are already using several addresses per each destination port to store the different available routing options.

To indicate that a given packet must be forwarded to a given destination port \( D \) by using a deterministic path, the first address \((d)\) assigned to the destination port will be used in the packet header, whereas if adaptive routing is desired, the address \((d + 1)\) will be used instead, regardless of the number of provided routing options. Thus, by inspecting only one bit (the least significant bit of the DLID), the switch knows if only one or all routing options must be provided. For instance, If the least significant bit of the DLID is not set, then the switch only returns the option stored at address \( d \). Of course, to make this scheme work, the path to be followed by packets sent using deterministic routing should be stored in the first address \((d)\) assigned to each destination port.

Notice that a given system may have both switches that support adaptive routing and switches that only support deterministic routing. There is no problem in combining both of them in the same network. Obviously, only the adaptive switches will provide more than one routing option. However, care must be taken to store, in the deterministic switches, all the table addresses that correspond to the same destination port with the same switch output port, the only available for that destination.

### 4.3 Selection of the Final Output Port

Once we have several routing options for the packet, one of them has to be finally selected. We will assume that original switch works as follows. The forwarding table is accessed as soon as a packet arrives at the switch, before reaching the head of the input buffer. After routing the packet, the returned output port is stored together with the packet and when it reaches the head of the buffer, arbitration for the output port is performed at the internal switch level.

In the enhanced switches, the selection can be done either immediately after output ports are returned by the forwarding table or can be delayed until arbitration is done at the internal switch. In both cases, the choice can be done without considering the status of the required output ports (for instance, randomly selecting the output port) or considering it (for instance, selecting the output port with more buffer space). Although making the selection at arbitration time [25] may lead to better performance as more updated status information can be used, it requires to store all the routing options obtained after accessing the forwarding tables until arbitration is performed. On the contrary, making the selection immediately after accessing the forwarding table is simpler, as it does not require to modify the switch arbitration logic. Notice that in this case output selection is performed according to the output port status available at that moment.

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

---

\(^3\)The subnet manager should guarantee that the APM mechanism uses different LIDs from those used for adaptive routing.
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 switch output port that corresponds to the first address is directly selected. Otherwise, which corresponds to adaptive routing, the selection is performed either immediately or at the internal switch arbitration time, and either considering the status of the output ports or performing a static selection.

### 4.4 Support for Adaptive Routing with Escape Paths

The next step to improve routing freedom is providing support for adaptive routing algorithms with escape paths in IBA. In particular, we are interested in the fully adaptive routing algorithm presented in Section 3, which requires the use of two queues per physical link, which are dynamically selected at routing time.

Although IBA supports virtual lanes (VLs) that could be used to implement these queues, it does not allow the selection of them at routing time. This is a serious limitation that prevents the use of FA routing. However, again, by using some tricks, we will be able to use this routing algorithm in IBA switches.

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 by doing this we are not using additional buffers nor additional VLs. Also, 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 the another one located in the middle of the physical VL buffer, 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. This can also be easily implemented when a RAM memory is used to implement the VL buffer, as packets stored in any memory location can be selected for routing.

On the other hand, by arranging the adaptive and escape queues in the same physical buffer (the only one visible to IBA specs), some packets initially stored in the escape queue may advance and enter the adaptive queue. This corresponds to a transition from an escape to an adaptive queue. As stated in Section 3, this does not lead to deadlock.

Whenever a new packet arrives at a switch, several routing options will be provided. In the case of FA routing algorithm, in all routing options but one, the adaptive queue of the VL at the next switch (reached through the output port) should be used. In exactly one, the escape queue of the VL at the next switch (reached through the output port) should be used. In IBA, next VL to be used is computed as a function of the input port, the output port, and the packet service level (SL) according to the SlottVL table [15]. In Section 4.1, we have already described a way of offering several different physical output ports by using virtual addresses and modifying the forwarding table access. Therefore, if we use only one adaptive routing option and one escape routing option, two addresses will be used for each destination port. The first one (d) will store the escape routing option, and the second one (d + 1) will store the adaptive routing option.

A critical issue in the mechanism is the selection of the final routing option to use. An improper selection of the output port could lead to deadlock. In particular, it must be guaranteed that the adaptive routing option is selected only if there is enough space in the adaptive queue associated to the corresponding VL. IBA uses a credit-based flow control mechanism and the credit information is computed on a per-VL basis. Given the maximum number of credits at a particular VL (C_{max}) and fixing the number of credits available to the escape queue (C_{0}), there is space in the adaptive queue of a VL if the available number of credits is greater than C_{0}. Otherwise, there is only space (if any) in the escape queue. In Figure 2, we can observe that the switch knows the available number of credits at each VL of every physical channel. In particular, it knows the available credits of VL0 belonging to output ports op0 and op1 (C_{00} and C_{01}), respectively. From these data, the switch can easily compute the available credits at the adaptive (C_{00A} and C_{10A}) and escape queues (C_{00E} and C_{10E}) for VL0. In particular, if the buffer associated to a VL is divided into two equally sized queues, then the number of available credits for the adaptive and the escape queue associated with output port X and VL Y is:

\[
C_{XY A} = \max(0, C_{XY} - (C_{max}/2)) \\
C_{XY E} = \min(C_{max}/2, C_{XY})
\]

where \(C_{XY}\) represents the credits available at VL Y of output port X.

Once the forwarding table offers the adaptive and escape routing options (leading to output ports op0 and op1 in...
Figure 1. Providing multiple routing options in IBA switches.

Figure 2) and the corresponding VL has been computed according to the SLtoVL table (VLO in Figure 2), the switch, based on the information about the number of credits ($C_{10A}$, $C_{00A}$ and $C_{00E}$ in Figure 2), can properly select the output port to use. The adaptive routing option can be used only if there are enough credits available ($C_{10A}$) at the adaptive queue. 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 ($C_{00A}$ and $C_{00E}$). Finally, in the case both routing options have no credits available for the packet, the switch can either decide to forward the packet through the escape queue (when there are enough available credits) or to take a decision later.

The selection of the final routing option must be done considering the status of the required output ports as explained previously, and ensuring that there is space at the adaptive queue when the packet is sent if the adaptive routing option is finally chosen.

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 stated in Section 4.2, the source node of a packet can also enable or disable adaptive routing by choosing the appropriate destination address in the packet header. When adaptive routing is disabled, the switch must return only one option, the escape routing option. Although these packets are forwarded through the escape queues, a “deterministic” packet may enter the adaptive queue as they are implemented in the same buffer. This packet will abandon the switch when it reaches the head of the buffer. As the escape queue has also a connection to the internal switch, some “deterministic” packets that arrive later might bypass other “deterministic” ones stored in the adaptive queue, which would compromise in-order delivery. The problem can be easily solved by using a pointer to the first “deterministic” packet stored in the adaptive queue. This packet must be forwarded before any other packet stored in the escape queue.

5 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 [15]. 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.

5.1 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 [5] (see Section 3), 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, as defined at Section 4.4.

Switches can support up to 16 virtual lanes. VLs can be used to form separate virtual networks. We will use a non-
In our simulations. And second, neighboring switches the same number of nodes connected to every switch (4
has the same number of ports (we used 8 or 10) and
First, we will assume that every switch in the network
per time unit.

The switch routing time will be set to 100ns, including
the time to access the forwarding tables, the crossbar arbirter

time, and the time to set up the crossbar connections.

Links in InfiniBand are serial. 10/8 coding [15] is used.
In the simulator, the link rate will be fixed to the 1X
configuration [15] (2.5 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 tried
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 latency4 measured in
nanoseconds versus the average accepted traffic5 measured
in bytes/ns/switch.

We will analize irregular networks of 8, 16, 32, and 64
switches randomly generated following some restrictions.
First, we will assume that every switch in the network
has the same number of ports (we used 8 or 10) and
the same number of nodes connected to every switch (4
in our simulations). And second, neighboring switches
will be interconnected by just one link. Ten different
topologies will be randomly generated for each network
size. Thus, we will also show minimum, maximum and
average results. In addition, we will also plot the results for
some representative topologies for every network size.

5.2 Evaluation Results
In this section we analyze the influence on network performance of using IBA switches with adaptive routing
capabilities when applying the FA routing algorithm. First,
we study the advantages of using IBA switches with
adaptive routing capabilities, also analyzing the influence of
the percentage of adaptive traffic on network performance.
Then, we analyze how the network connectivity and the
degree of freedom provided by the switches affect network
performance.

5.2.1 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, whereas the four links 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, 75% of
adaptive traffic almost obtains the same network throughput
than 100% of adaptive traffic. On the contrary, for the
64-switch network, the difference in network throughput
when injecting 75% and 100% of adaptive traffic is greater.
When using 100% of adaptive traffic, network throughput
is noticeably increased, ranging the factor of throughput

---

4Latency is the elapsed time between the generation of a packet at the source
host until it is delivered at the destination end-node.
5Accepted traffic is the amount of information delivered by the network
per time unit.
increase from 1.2 for the 8-switch network to 3.33 for the 64-switch network.

Table 1 shows minimum, maximum, and average factors of throughput increase for different network sizes and different packet sizes when using 100% of adaptive traffic. As we can see, network throughput increase becomes higher as network size increases. In particular, for 32-byte packet uniform traffic pattern, when using networks with two routing options and 4 links connecting switches, network throughput is increased, on average, from 1.5 to 3.27, 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 [9]. Hence, packets benefit more from using adaptive routing.

For the hot-spot traffic, lower throughput increases are achieved. As it is shown in Table 1, the higher the percentage of applied hot-spot traffic, the lower the factors of throughput improvement. This is because traffic tends to concentrate around the hot-spot host, spreading the congestion over the entire network as all the hosts send packets to the hot-spot, preventing other packets from taking advantage of using adaptive routing.

Indeed, for the bit-reversal traffic pattern (which creates some local congestion areas), results similar to the uniform traffic pattern are obtained, as shown in Table 1. 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.

5.2.2 Influence of Increasing Routing Options and Network Connectivity

The benefits of using adaptive routing increase as the number of routing options available at each switch port increases. Also, the higher the network connectivity, the greater the benefits of using adaptive routing. Table 2 shows the average percentage of routing options per destination at each switch for different network topologies. As can be observed, as network connectivity increases, the percentage
of destinations with more than one routing options is increased.

With a connectivity of 4 links per switch, the maximum number of routing options per switch would be theoretically four. However, as can be observed in Table 2, in most cases it is not worth providing more than two or three routing options per switch. For instance, for a 64-switch network, more than two options are provided for only 17.48% of the destinations.

However, when using 6 links to connect to other switches we can observe that the percentage of cases where there are more than two available routing options increases. For 64-switch networks, on average, the percentage of cases where there are more than two routing options is 30%

Table 1 (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 (3.27 vs 3.47 for 64-switch networks and 32-byte packets). However, as network becomes more connected (with 6 links connecting switches), higher throughput is obtained as more routing options are allowed. Particularly, throughput increases by a factor of 3.90 for 32-byte packets with uniform traffic and 64-switch networks when up to four routing options are allowed. Despite it, these evaluation results also show that only two routing options are enough to obtain roughly 90% of the maximum throughput improvement.

6 Conclusions

The main contribution of this paper is the proposal of 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. An interleaved memory organization is used to implement linear forwarding tables at each switch while allowing simultaneous accesses to them. 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 [5]. Results show that enhancing IBA switches with adaptive routing noticeably increases network performance. This is specially significant for large networks. Also, 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 [10, 11] 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.
Table 2. Average percentage of routing options at each switch for each destination port. MR stands for Maximum number of Routing Options at each switch for each destination.

<table>
<thead>
<tr>
<th>Sw</th>
<th>MR</th>
<th>4 links</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>57.14</td>
<td>22.14</td>
<td>20.72</td>
<td>1.26</td>
<td>49.33</td>
<td>24.33</td>
<td>26.34</td>
</tr>
<tr>
<td>16</td>
<td>2</td>
<td>54.79</td>
<td>45.21</td>
<td>31.75</td>
<td>13.46</td>
<td>49.33</td>
<td>24.33</td>
<td>26.34</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>54.79</td>
<td>31.75</td>
<td>11.83</td>
<td>1.62</td>
<td>49.33</td>
<td>24.33</td>
<td>26.34</td>
</tr>
</tbody>
</table>

| 64 | 2  | 48.07   | 51.93    | 35.86    | 16.07    | 45.39    | 26.49    | 28.12    | 16.07    |
| 64 | 4  | 48.07   | 35.86    | 13.21    | 4.59     | 45.39    | 26.49    | 13.50    | 14.62    |

| 64 | 3  | 41.32   | 58.68    | 41.20    | 17.48    | 37.29    | 62.74    | 30.06    | 19.13    |
| 64 | 4  | 41.32   | 41.20    | 14.09    | 3.39     | 37.29    | 62.74    | 19.13    | 10.93    |

References


