Design and Evaluation of Scalable Switching Fabrics for High-Performance Routers

Nian-Feng Tzeng and **Ravi C. Batchu**

Center for Advanced Computer Studies
University of Louisiana at Lafayette
Lafayette, LA 70504, USA

Abstract

This work considers switching fabrics with distributed packet routing to achieve high scalability and low costs. The considered switching fabrics are based on a multistage structure with different re-circulation designs, where adjacent stages are interconnected according to the indirect n-cube connection style. They all compare favorably with an earlier multistage-based counterpart according to extensive simulation, in terms of performance measures of interest and hardware complexity. When queues are incorporated in the output ports of switching elements (SE's), the total number of stages required in our proposed fabrics to reach a given performance level can be reduced substantially. The performance of those fabrics with output queues is evaluated under different "speedups" of the queues, where the speedup is the operating clock rate ratio of that at the SE core to that over external links. Our simulation reveals that a small speedup of 2 is adequate for buffered switching fabrics comprising 4×8 SE's to deliver better performance than their unbuffered counterparts with 50% more stages of SE's, when the fabric size is 256. The buffered switching fabrics under our consideration are scalable and of low costs, ideally suitable for constructing high-performance routers with large numbers of line cards.

1. Introduction

Rapidly growing demand for high-speed networks has prompted the investigation into scalable routers that are capable of forwarding data at the aggregate rate of multi-terabits per second. Such a router contains many line cards (LC's) for admitting external links of various speeds. Those LC's are interconnected by a switching fabric to provide paths for packets to travel from arrival LC's to their respective departure LC's. Routers serve to connect external links with various data rates through its line cards (LC's). On receiving a packet, an LC transmits it, based on the packet destination, to the outgoing LC through which the packet is to be delivered forward. The outgoing LC is determined by a table lookup, performed either locally at the arrival LC or via a remote forwarding engine (FE). Any router with multiple LC's (common in a current high-performance router) employs a switching fabric to interconnect its constituent LC's, providing paths among LC's for packets to move from their arrival LC's toward their destined LC's. Switching fabrics naturally affect overall router performance and dictate router scalability.

It is highly desirable to devise scalable fabrics for future high-performance routers which may contain large numbers of LC's. A multistage switching structure is referred to as the space-division architecture and comprises multiple stages of switching elements (SE's) with or without buffers. Different multistage-based configurations have been introduced, and the Shuffleout has been shown to outperform others [1]. A variation of the Shuffleout, formed by providing paths from its primary outputs back to its primary inputs, was considered. It is known as the closed-loop Shuffleout [2], which allows for packets to be re-circulated back, when they fail to reach their destinations at the primary outputs. Both the Shuffleout and the closed-loop Shuffleout are unbuffered. An unbuffered multistage structure enjoys the advantage of hardware simplicity.

Buffers can be introduced to SE's in a multistage structure to enhance its throughput, because they can hold packets which head for the same output port simultaneously and which otherwise have to be dropped or deflection routed [7]. Packets held in the buffers are delivered in sequence later on. For a multistage-based switching fabric, incorporating buffers in its constituent SE's can lower the number of stages required to achieve a given performance level. Buffers may be incorporated in the output (or input) ports of SE's, forming output (or input) queues. It has been shown that output queuing outperforms input queuing [5], but output queuing requires a "speedup," where the core of SE's runs faster than the connected links. Output queuing does not suffer from the *head-of-line* problem. Previous simulation and analytical studies indicated that a small speedup (say, 2 – 4) was enough in practice for output queuing to deliver packets quickly to their output ports [4].

In this article, we propose and evaluate switching fabric designs, dubbed the *I-Cubeout* [11], for scalable routers. Our proposed fabrics comprise multiple stages of SE's interconnected following the indirect *n*-cube connection style [9], with different approaches for re-circulating packets from their primary outputs back into the fabrics. The

This work was support in part by the National Science Foundation under Grants EIA-9871315 and CCR-0105529, by the Army Research Office under Grant/Cooperative Agreement No. DAAG55-98-1-0240, and by the Board of Regents of the State of Louisiana under Contract No. LEQSF(2000-2001)-ENH-TR-90.

simplest re-circulation design requires little control logics for re-circulating paths but exhibits relatively lower performance. Other two approaches make use of more control logics to arrive at better performance. Our simulation has unveiled that the proposed switching fabrics all outperform their compatible closed-loop Shuffleout. When buffers are introduced to output ports of SE's, it is found that the number of stages for our proposed fabrics to yield a given performance level is reduced substantially, even for the speedup of only 2. Since the costs of our switching fabrics are likely to be proportional to the number of stages involved, buffered fabrics appear more attractive than their unbuffered counterparts and are better applicable to large sized routers with hundreds of LC's.

2. Related Work

An I-Cubeout switch with size N (= 2^n) is denoted by ICO_N, where each SE is a small crossbar of size $b\times 2b$ to provide b outlets for terminating packets at their destination queues as soon as they reach their destined rows [11]. For a 2×4 SE (i.e., b = 2), it has two remote outlets (for connecting SE's in the next stage) and two local outlets (for terminating packets at destination queues). Adjacent stages are interconnected according to an indirect n-cube connecting pattern [9]. Specifically, the two remote outlets of any SE in the same stage differ in their addresses by a constant; those in stage 1 (i.e., the leftmost stage) differ by N/2, those in stage 2 differ by N/4, and so on, for ICO_N. At the n^{th} stage, the two remote outlets of any SE differ by 1. A *copy* of the indirect n-cube connection spans from stage 1 to stage n. In stage n+1, the two remote outlets of an SE differ by N/2, identical to the situation of stage 1. This stage begins another copy of the indirect n-cube connection, which covers the next n stages. ICO_N may contain any number of stages, not necessary an integer multiple of n. For example, ICO_8 may contain five (5) stages, while a copy of the indirect 3-cube covers three (3) stages.

Packets in ICO are self-routed in a distributed manner, with the routing tag of each packet computed at the primary input according to its source and destination addresses via a bit-wise XOR operation [11]. The tag is carried along with the packet to guide its traversal across ICO. The first tag bit is used by SE's in Stage 1 of any copy, and the second tag bit is by SE's in Stage 2 of any copy, etc. If a tag bit, say in position p, is "1", the packet takes a "cross" state at the visited SE in stage p of any copy, signifying that the upper (or lower) inlet of the SE is connected to the lower (or upper) remote outlet. It reflects that the packet is "corrected" at position p, and the correction can be done in stage p of any copy. After it is done, the p^{th} tag bit is reset to "0". If the tag bit is "0", the visited SE is set to the "straight" state, since no correction is then needed. After a packet advances one stage, its tag is rotated leftward by one bit position, so that the tag bit to be examined at any stage is always the leftmost one. When the tag bits all become "0", the packet has reached its destined row and may take the associated local outlet to reach its destination queue.

If two packets at an SE in stage p have distinct values in their p^{th} tag bits, a conflict occurs and only one of the two conflicting packets is forwarded to its destined outlet, with the other being deflection-routed [11]. This is because the SE can be set to either the cross or the straight state at a time, but not both. The packet with a smaller distance to its destination is given priority, where the distance is defined as the minimum number of stages a packet has to travel before getting to its destination. The distance of a packet can be told directly from its current tag. A simple priority resolution logic is devised for this purpose [11]. For SE with output queues and with a speedup of k, up to k competing packets are accepted by any output queue and deflection-routing happens only if there are more than k packets competing for one output port or if the targeted output queue has a capacity less than what is needed to hold k packets.

While ICO explained so far is composed of 2×4 SE's, it is obvious that ICO in general can be built by $b\times 2b$ SE's, each connecting to b SE's in the next stage and terminating at b destination queues. When a packet has not reached its destined row at an SE, it is forwarded to the next stage through the remote outlet decided by the leftmost $\log_2 b$ routing tag bits. If a packet fails to reach its destined row after traversing all the stages, the packet is dropped at the output side of the last stage (called the *primary output*s). It is found that ICO exhibits lower hardware complexity than its (open-loop) Shuffleout counterpart [11] mainly because every SE in the Shuffleout switch is equipped with b *distance computing blocks*, one for each input. The complexity of such a computing block grows in the order of approximately $O(m^3)$, where m equals $\log_b N$ for a switching fabric of size N comprising $b\times 2b$ SE's [3]. This rapid growth in hardware complexity makes it prohibitively expensive to construct a large Shuffleout switch, restricting its scalability.

The closed-loop Shuffleout [2] was formed by adding re-circulation paths to the open-loop structure so that packets may travel through all the stages several times until they reach their destined output queues eventually (or get dropped when storage at each primary input for holding re-entered packets is fully exhausted), where a *primary input* is an input of the first stage. Storage is provided at each primary input for holding concurrently arrived packets, but there is no queue at the input or output ports of its constituent SE's.

3. Proposed Switching Fabrics — NONBUFFERED CASES

Our proposed switching fabrics without buffers for scalable routers are based on ICO with appropriate re-circulating connections from the primary outputs back to the fabrics. These connections enable the packets to re-enter the fabrics when they

fail to reach their destined rows in the primary output side, instead of being dropped otherwise. They aim to lower the number of stages required in ICO for a given performance level, when compared with ICO without such re-circulating connections, in order to simplify the destination queues logics and to reduce the total number of stages (of SE's). For efficient utilization of fabric resources, the re-circulating connections in ICO are to be fed to the *last* copy of fabrics either statically or dynamically, unlike the closed-loop Shuffleout which re-circulates connections back to the fabric primary input side statically. Our ICO's all lead to fewer conflicts and thus less resource wastage when using the last copy of stages for routing re-entered packets to their destined rows. This is because traffic is consistently lighter in a later stage of ICO_N, and one full copy (of $\log_b N$ stages) is needed to route any packet to its destination. From this point onward, a packet is designated as a *fixed-length data unit* which can be delivered from one SE to another SE in a subsequent stage in one cycle. Messages are of varying sizes and are partitioned into packets (of fixed length) before being injected into switching fabrics for delivery. At their destinations, packets are put back to their original messages before being processed at the outgoing line cards (e.g., producing appropriate headers and performing needed fragmentation according to the protocols employed therein). Different re-circulation designs for ICO's are described next.

A. Static Re-circulating Connections

The simplest form of re-circulation is realized by connecting each primary output (at the last stage) back to the input of $\log_b N$ stages ahead in the same row, for ICO_N composed of $b\times 2b$ SE's. It has a *static* re-entry point for each re-circulating connection, denoted as ICO_N^S, irrespective of whether or not there is a packet arriving from a prior stage at that re-entry point. ICO_8^S with five stages of 2×4 SE's is illustrated in Fig. 1, where the re-entry point of each re-circulating connection is 3 stages ahead (of the primary output side). Making re-circulating connections this way involves the least amount of extra logics. Each re-entry point is equipped with a 2-to-1 multiplexor, denoted by "M" in the figure, to accommodate the re-circulating connection.

The routing tag of each packet sent along a re-circulating connection back to the switching fabric *need not* be recomputed or modified, as the re-circulated packet is treated as if it advances to a next stage. There is no buffer provided at each re-entry point, and the packet can be fed through a re-circulating connection back to ICO^S only if there is *no packet arriving* from the prior stage to that re-entry point concurrently. This means a re-entered packet has lower priority, and resources are made available to re-entered packets only if they are otherwise not utilized. Note that if re-circulated packets are fed back to the first stage of fabrics, like the closed-loop Shuffleout, all such packets are dropped when the load equals 1.0 where a new packet is always injected into every primary input in each cycle, causing the re-circulated packets (which are older) to be discarded. After a packet gets re-entered into the fabric but conflicts with another packet later at any SE (in the last copy of $\log_b N$ stages), the one closer to its destination is given priority, irrespective of its age. If their distances are identical, the older packet (i.e., the re-entered one) gets priority. This is so desired to keep the *worst packet latency* small in ICO with re-circulating connections and is the only additional logic added in SE's in the presence of re-circulating connections.

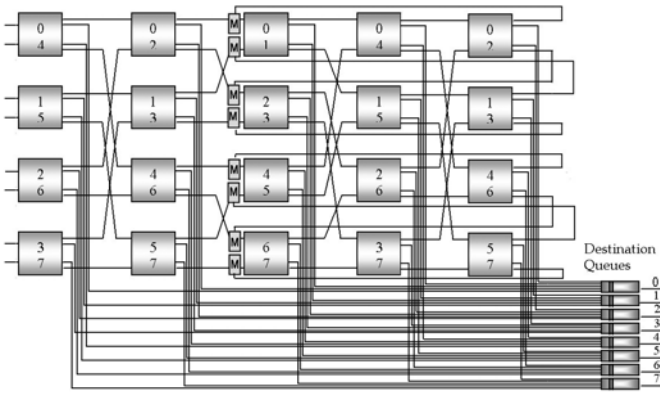


Fig. 1. ICO_8 with static re-circulating connections, ICO_8^S.

B. Dynamic Re-circulating Connections – First Avail

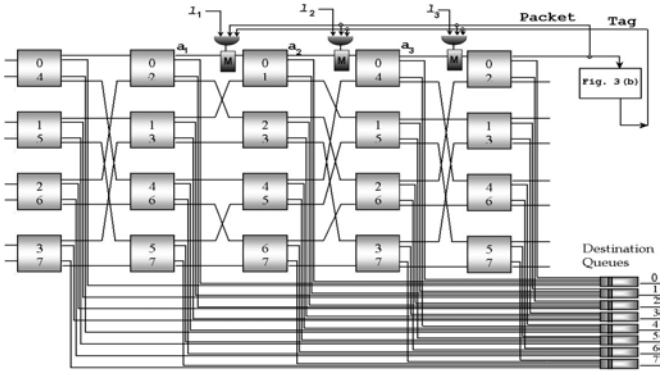
In order to lower the possibility of dropping re-entered packets due to conflicting with other concurrent packets from the prior stage, every re-circulating connection is provided with $\log_b N$ entry points, one for each stage of the last copy, as shown in Fig. 2(a). Each entry point is equipped with a 2-to-1 multiplexor to accommodate a re-circulating connection. When a packet is to re-enter the fabric, it utilizes the entry point which corresponds to the *first* available inlet, where an inlet is available (at the time when a packet is to re-enter the fabric) if the prior stage delivers no packet to the inlet simultaneously. As the outlet of each SE has a latch to hold one packet at the end of each cycle, the availability of an inlet can be told directly using the latch indicator of the outlet (in the prior stage) which is connected to the inlet. If none of the inlets in the last copy is available, the packet is dropped. This fabric design re-circulates packets back via the first available entry points dynamically, referred to as ICO^{FA}. Such a design is expected to get more packets re-circulated successfully back to the fabric at the expense of more multiplexors and their control logics.

Consider a given re-circulating connection. Let the availability of an inlet at stage i of the last copy be denoted by a_i (with $a_i = 1$ signifying that the inlet is available), the control signal for loading the packet via the entry point at stage i along the re-circulating connection shown in Fig. 2(a) is expressed as

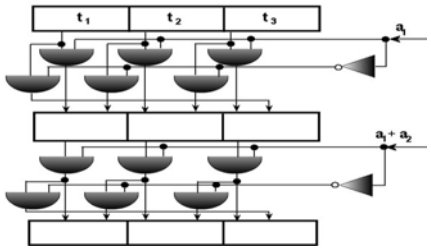
$$l_i = \underline{a}_1\underline{a}_2 \ldots \underline{a}_{i-1}a_i,$$

where $\underline{a}_j$ represents $-a_j$ for $1\le j\le i-1$. Note that Fig. 2(a) provides only the top re-circulating connection for clarity, leaving out the remaining connections. These control signals

ensure that a re-circulated packet re-enters the fabric through the first available entry point (and only that one). When a packet is fed back at stage f, its tag has to be rotated leftward by $(f\text{-}1)\times\log_2 b$ bit positions before re-entry. A fast, simple logic to achieve such tag rotation is depicted in Fig. 2(b). The logic involves $\log_b N$ registers of length $\log_2 N$, with an appropriate control signal for data paths between a pair of adjacent registers, as given in the figure. After the packet re-enters the fabric with its proper tag accompanied, it is routed following the same routing algorithm. If it conflicts with another packet in an SE, the associated priority resolution logic is invoked to decide the one to be routed to its desired outlet, with the other being deflection-routed.



(a)



(b)

Fig. 2. (a) ICO_8 with dynamic re-circulating connections, ICO_8^{FA}, (b) Tag rotation logic for a re-circulating connection.

In general, ICO^{FA} does not have to provide entry points for *all* stages in the last copy, but just the early few stages, to restrict hardware overhead with almost no impact on performance measures of interest. It is found by our simulation that providing three (a constant number of) entry points is almost as good as providing $\log_b N$ entry points for any ICO_N^{FA}, presenting good scalability. We thus refer to ICO_N^{FA} as the design with *three entry points* per re-circulating connection (which translates to constant hardware overhead) subsequently. Note that ICO_N^{S} may be viewed as a special case of ICO_N^{FA} where only one entry point is provided for each re-circulating connection; in this case, no tag rotation is necessary.

The additional two entry points along each re-circulating connection in ICO_N^{FA} enhance throughput (or equivalently, offered load) noticeably, in comparison to that of ICO_N^{S}. In this article, the terms of throughput and offered load are used interchangeably. The throughput gain is more pronounced when the total number of stages is smaller. Simulation results of ICO^{FA} with different total stages will be presented in the next section. For the cases of buffered ICO (as will be treated in the next section), however, ICO_N^{FA} is found to exhibit only negligibly better performance than ICO_N^{S}.

C. Dynamic Re-circulating Connections – First "1" Bit

While ICO^{FA} provides three entry points per re-circulating connection and allows a packet to re-enter the fabric at the first available stage, it does not check if the routing bit(s) corresponding to that re-entry stage is (are all) "0". This is not the best point for the packet to re-enter the fabric, unless the routing bit(s) for that stage is not (are not all) "0". The reason is two-fold: (1) the packet stays in the same row after that stage if the routing bit(s) is (are all) "0", and having the packet re-enter the fabric there costs one extra cycle unnecessarily, and (2) the re-entered packet may conflict with another packet in the re-entry stage and get deflection-routed, taking an extra (otherwise unneeded) re-circulating trip to "correct" this deflection-routing. The throughput gain out of ICO^{FA} may be offset somewhat by an increased mean latency.

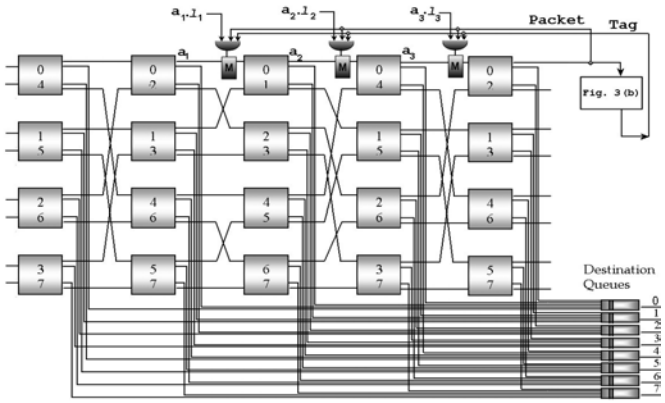


Fig. 3. ICO_8 with dynamic re-circulating connections, ICO_8^{FO}.

The best entry point for a re-circulated packet obviously corresponds to the *first* "1" bit in its routing tag, dubbed ICO^{FO}. This is because the packet has to get "corrected" in that corresponding stage before it can reach its destined row. To this end, an entry point is provided at every stage of the last copy for each re-circulating connection, as demonstrated in Fig. 3. The control signal for the entry point at stage i shown in the figure is given by

$$l_i = \underline{t}_1\underline{t}_2 \ldots \underline{t}_{i-1}t_i,$$

where $\underline{t}_j$ (or t_i) is the 'NOR' (or 'OR') result of the routing bit(s) for stage j, $1 \le j \le i\text{-}1$ (or stage i). If the re-entered packet happens to conflict with another packet at the entry point, the former packet is dropped. This is done simply by making use of the availability indicator of an inlet at stage i

of the last copy, a_i, as shown in Fig. 3. If the packet re-enters the fabric via the entry point of stage *f*, its tag is to be rotated leftward by $(f\text{-}1)\times\log_2 b$ bit positions before re-entry, using the same logic as depicted in Fig. 2(b), with control signals produced using t_j rather than a_j, for $1 \le j \le 2$. After a packet re-enters the fabric, it is routed in the same way as that described for ICO^{FA}. Unlike ICO_N^{FA}, ICO_N^{FO} requires *exactly* $\log_b N$ entry points for each re-circulating connection, since the first "1" tag bit may correspond to the last stage (of the last copy). As ICO^{FO} selects the best entry point for each re-entered packet, it is expected to exhibit the highest performance measures of interest for a given number of total stages, most suitable (among the three designs) for scalable router construction.

4. Buffered Switching Fabrics

Buffers are commonly introduced to the SE's of switching fabrics for performance improvement, in addition to their needs in packet switches for offering quality-of-service guarantees by establishing separate queues at each switch port for different traffic flows with various priority levels [10]. Buffers can be placed at the output (or input) ports of SE's to constitute output (or input) queues. Output queuing is known to exhibit better performance and be spared from the head-of-line blocking problem, but it often requires a "speedup" to achieve its peak throughput (of 100% potentially), where a speedup refers to that the switch core runs faster than the external links, so that multiple packets competing for an output port can be accepted by its associated output queue in one cycle [5]. In this work, we intend to explore the potential savings in total numbers of stages (of SE's) resulting from incorporating output queues in our proposed switching fabrics. This savings translates to cost reduction because the cost of a multistage-based fabric is largely proportional to the total number of stages (which dictates the overall chip count). In addition, few stages make the concentration logics in front of each destination queue simpler, further lowering the hardware cost.

Let the speedup at an output queue (of SE's) be denoted by ζ. For a fabric constructed out of $b\times 2b$ SE's, ζ is clearly no greater than ζ since there are at most ζ packets competing for an output queue (at an SE) in one cycle. While the number of lines terminating at each destination queue for a proposed fabric is equal to the number of stages in the fabric, a simple selector is placed in front of each destination queue in order to choose up to ζ packets in each cycle for acceptance by the queue, and those packets chosen are from the *last* ζ *active* stages (with respect to the destination), where an active stage has a packet to be delivered to the destination queue. More details about the selector design are provided in the next section. Destination queues are thus assumed to have a speedup of ζ in our switching fabrics. As will be illustrated by the simulation results later, performance of all the proposed fabrics is sensitive to ζ, and an increase in ζ leads to better performance for any given ICO_N^S, ICO_N^{FA}, or ICO_N^{FO}. When a fabric is composed of 4×8 SE's, it is possible to run at a rate of 200 MHz, which is deemed rather moderate with the current manufacturing technology (given that the spider chip which employs a full 6×6 crossbar is operating at 200 MHZ [12]). With this clock rate, ζ may be pushed to 4, when each output queue takes one packet in 1.25 *ns*. This is realizable practically, since the current on-chip SRAM (static RAM) can have an access time as little as 1 *ns*.

The routing procedure in SE's with output queues is identical to that explained in Section II for SE's without queues, except for deflection-routing decision. Specifically, in a $b\times 2b$ SE with the speedup of ζ ($\le b$) at output queues, if there are more than ζ packets heading for an output port in one cycle, the associated output queue can accept up to ζ such competing packets in a cycle, provided that the queue has capacity to hold them. Note that each queue has a specified capacity which can hold a number of packets (of fixed length). In each cycle, the packet, if any, at the head of each queue is moved to the next stage, say stage *p*, as follows. If the packet has reached its destined row in stage *p*, it is sent to the queue associated with the local port connected to its destination queue (referred to as a local queue), provided that the queue has capacity to take it and there are no more than ζ packets competing for the same queue at the same cycle. If the queue has no capacity left, the packet is sent to the queue associated with the remote output port connecting to the same row (called a remote queue) in stage *p+1*; the packet then attempts to reach its destined local queue in stage *p+1* during the next cycle. If there are more than ζ competing packets, ζ of them are randomly chosen for reaching the destined local queue, with the rest directed to the associated remote queue (for delivery to their destined local queue in the next stage).

If a packet has not reached its destined row in stage *p*, it is forwarded to a remote queue determined by the routing bit(s) of the packet. When there are more than ζ packets competing for the remote queue, the distance of each packet dictates if the packet is to be sent to the desired queue or to be deflection-routed: the packet which is closer to its destination is given priority, like in the nonbuffered cases. The priority resolution logic devised for the nonbuffered SE's [11] can be employed for this purpose after a minor modification. For the speedup of ζ, the SE core is operating ζ times faster. However, the resolution logic, being a simple combinational circuit, in an SE can arrive at one decision in less than $(1/\zeta)$ cycle time (called a phase); for example, with the cycle time of 5 *ns* and $\zeta = 2$, we have $(1/\zeta)$ cycle time equal to 2.5 *ns* and the resolution logic is expected to produce one decision in a fraction of 2.5 *ns*. After a packet is chosen as a winner, the logic input corresponding to the winner packet is changed to all "1" (reflecting a farthest distance) so that the packet with the

second smallest distance is selected in the next decision phase. This process repeats ζ times in a cycle to choose ζ winners for reaching the desired queue. All losers, if any, are deflection-routed to other remote queue(s) in the SE.

5. Performance Evaluation

Different types of $b \times 2b$ SE's for building various sizes (N) of proposed switching fabrics have been examined via simulation, including b = 2, 4, 8 and N = 64, 256. As the simulation results point to a similar trend, this section presents only the results of switching fabrics comprising 4×8 SE's with N = 256. We illustrate and discuss the results for unbuffered switching fabrics first, followed by those for switching fabrics with output queues operating under different speedups. As will be seen, a buffered switching fabric requires much fewer stages to deliver a given performance level than its unbuffered counterpart, even when the speedup is only 2.

A. Simulation Model

A copy of such a fabric composed of $\log_4(256) = 4$ stages. Each primary input is assumed to generate packets of fixed length randomly with their destinations uniformly distributed between 0 and 255 under different loads. A packet moves from one SE to another in the next stage in one cycle, based on the routing algorithm. A re-circulated packet in any of the three fabric designs also takes one cycle to get re-admitted. Each data value given in Figs. 4-7 is the result after 200,000 clock cycles in our simulation, where this number of cycles is found to yield steady-state outcomes. The performance measures of interest include *mean latency*, *offered load* (or *throughput*), and *packet drop rate*. Mean latency signifies the average number of cycles for a packet to reach its destination queue after it is generated. The number of packets arriving at a typical destination queue per cycle is defined as offered load (or throughput), whereas the probability of a packet gets dropped in the switching fabric under a given load reflects the packet drop rate. For buffered switching fabrics, each SE output queue (either local or remote one) is equipped with a buffer for holding 12 packets. When an output queue runs out of its capacity (i.e., unable to keep those selected ζ packets in one cycle, where ζ is the speedup), some packets which otherwise should reach the queue are deflection-routed. Each queue has a bypass provision such that a packet entering the queue does not have to take 12 cycles to exit unless there are 11 packets situated in front of it. This provision allows a packet which enters an empty queue to leave the queue immediately in the next cycle. A small queue capacity degrades performance for the fabrics under consideration, in particular, when ζ is 4; on the other hand, an excessively large queue does not enhance performance noticeably. As a result, we present only results for the queue capacity equal to 12 slots (i.e., packets) here.

B. Results and Discussion – Unbuffered Cases

Let k denote the total number of stages involved in a switching fabric. The mean latency versus offered load for k = 7 under ICO_{256}^{S}, ICO_{256}^{FA}, and ICO_{256}^{FO} is depicted in Fig. 4. The performance results of a compatible closed-loop Shuffleout (denoted by CSO_{256}) are also included for comparison. As can be observed, ICO_{256}^{S} under k = 7 exhibits considerably better performance than its CSO_{256} counterpart. For a given offered load, ICO_{256}^{S} enjoys sizable reduction in mean latency, by up to almost 20%. The maximum sustained throughput is slightly higher under ICO_{256}^{S} than under CSO_{256} for k = 7. This performance advantage is achieved, even though the constituent switching elements are less complicated in ICO than in CSO, as elaborated in Section II. It results directly from the fact that re-entered packets in ICO travel through the last 4 stages (i.e., last copy) where traffic is far lighter than that in earlier stages. When k grows, the performance gaps shrink, as can be found for the case of k = 10 in Fig. 4, since fewer packets then need to be re-circulated.

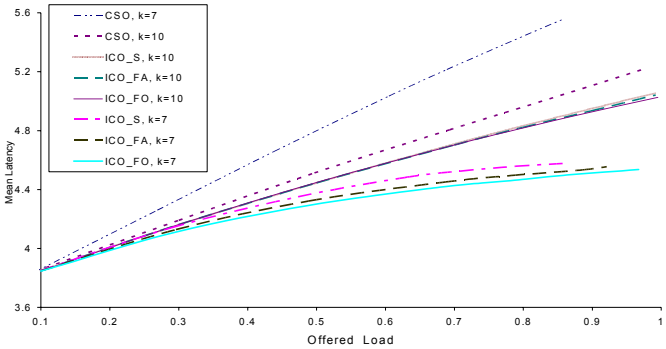


Fig. 4. Mean latency versus offered load for k = 7 and k = 10.

The re-circulation designs are expected to have substantial impacts on overall performance, in particular, for a relatively small k, say, $k < 2*\log_b N$. When k equals 7, for example, ICO_{256}^{S} achieves the maximum offered load of 0.87 only, whereas ICO_{256}^{FO} can sustain the offered load of 0.97. In addition, for any given throughput, ICO_{256}^{FO} results in slightly lower mean latency than the other two designs, for both k = 7 and k = 10. From the performance standpoint, ICO_{256}^{FO} is clearly superior. While ICO_{256}^{FO} requires hardware for rotating the tag of every re-entered packet properly along each re-circulating connection (as described in Section III), it takes fewer stages when compared with ICO_{256}^{S} to achieve the same performance level. For example, ICO_{256}^{FO} needs only 7 stages to yield the maximum offered load achievable by ICO_{256}^{S} with k = 9 (derived from the results in Fig. 5 below). These savings in the reduced stage count and in simplifying interfaces to the destination queues more than offset the added hardware amount, since four pieces of tag rotation logics (needed for the four outlets of an SE in the last stage) are clearly far simpler than a 4×8 unbuffered SE (consult Fig. 2(b) for a tag rotation logic). On

the other hand, ICO_{256}^{FO} possesses slightly more hardware than ICO_{256}^{FA} but delivers a marked gain in the maximum offered load for $k = 7$.

The switching fabric of a router often employs a back-pressure mechanism to refrain packets from entering the fabric once its resources are exhausted [6, 8], instead of dropping packets after they are admitted. When no back-pressure mechanism is adopted, the drop rate performance measure serves as a good indicator of the probability that the backpressure mechanism is engaged. The packet drop rate versus k under the load of 1.0 is demonstrated in Fig. 5. For any given k, ICO_{256}^{S} is found to enjoy consistently a smaller drop rate than its CSO_{256} counterpart, as expected. Likewise, ICO_{256}^{FO} outperforms ICO_{256}^{S} and ICO_{256}^{FA} for all k values examined. The gap between the drop rate of ICO_{256}^{FO} and that of ICO_{256}^{S} (or ICO_{256}^{FA}) is particularly large for $k < 8$. This is mainly because the switching fabric with $k < 8$ does not have a full copy of stages to route packets without competing with re-entered ones (since less than two full copies of stages are involved), making it especially crucial to conserve fabric resources by admitting re-circulated packets only through their best entry points, as done by ICO_{256}^{FO}. When k grows, fewer packets are re-circulated and the performance gain due to ICO_{256}^{FO} decreases accordingly, as shown in Fig. 5.

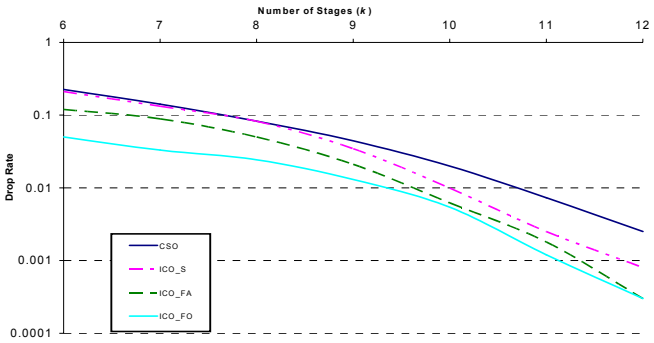


Fig. 5. Packet drop rate versus k under the load of 1.0.

C. *Results of Buffered Switching Fabrics*

A buffered switching fabric is expected to exhibit better performance than its unbuffered counterpart for any given number of stages of SE's. In Fig. 6, mean latency versus offered load for the three buffered switching fabrics is demonstrated under $k = 4$ and $k = 6$ with the speedup of $\zeta = 2$. It should be noted that the minimum number of stages of 4×8 SE's required for any multistage-based fabric of size $N = 256$ is 4 (referring to as one copy of stages earlier). With $k = 4$, ICO^{S} exhibits the maximum offered load of 0.85, almost identical to what ICO^{FA} does. In addition, ICO^{S} and ICO^{FA} are found to yield roughly the same mean latency throughout the entire range of the offered load, signifying that they offer almost identical performance (despite more complicated re-circulating connections for ICO^{FA}). On the other hand, ICO^{FO} reaches the maximum throughput exceeding 0.98 for $k = 4$, and its corresponding performance curve is noticeably lower than those of ICO^{S} and ICO^{FA}. This indicates that ICO^{FO} clearly outperforms its two counterparts when buffers are introduced to SE's, like unbuffered cases demonstrated previously. If the number of stages grows to 6 (i.e., $k = 6$), ICO^{S} and ICO^{FA} again exhibit almost identical performance, with their maximum offered loads beyond 0.98, as illustrated in Fig. 6.

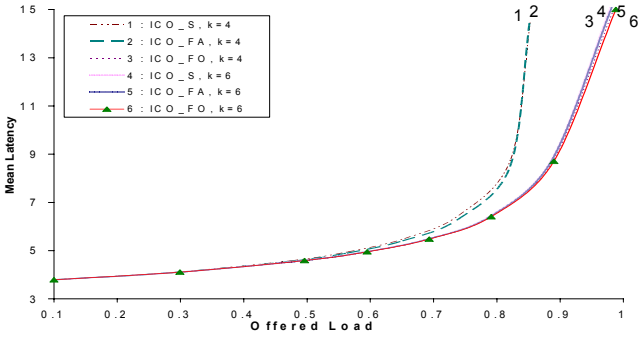


Fig. 6. Mean latency versus offered load for buffered switching fabrics under $k = 4$ and $k = 6$ with $\zeta = 2$.

Under this speedup (of $\zeta = 2$), it appears that $k = 6$ is adequate for ICO^{S} and ICO^{FA} to sustain satisfactory performance levels. As expected, ICO^{FO} again exhibits better performance than ICO^{S} and ICO^{FA}, but the performance gap shrinks in this case than in the case of $k = 4$. The maximum offered load for ICO^{FO} with $k = 6$ approaches 0.99. For ICO^{S} and ICO^{FA}, the increase of k from 4 to 6 substantially enhances its performance, whereas for ICO^{FO}, the performance improvement amount is negligible since its performance under $k = 4$ is very good already. In general, ICO_N^{FO} is a superior design choice among the three buffered switching fabrics (composed of $b \times 2b$ SE's), in particular when k is equal to, or only slightly more than, $\log_b N$.

The packet drop rate as a function of k is shown in Fig. 7 for the three switching fabrics with different speedups (ζ) under the load of 1.0. Under a speedup of $\zeta = 2$, all switching fabrics experience reduced drop rates as k grows. The drop rate reduction accelerates when k goes beyond 6, because the re-circulated packets not only are few but also are fed back into the fabrics through a stage whose traffic is expected to be very light. For any k, buffered ICO^{FO} always maintains a smaller drop rate than its ICO^{S} and ICO^{FA} counterparts. If the speedup is pushed up aggressively to $\zeta = 4$, the drop rates for the three switching fabrics not only are consistently much lighter for any k but also go down quicker as k increases, when compared with those under $\zeta = 2$. As might be expected, ICO^{FO} is observed to outperform its two counterparts by a wide margin. Under $k = 6$, for example, ICO^{FO} gives rise to a drop rate less than 10^{-4}, in contrast to 10^{-3} exhibited by ICO^{S} and ICO^{FA}. When k exceeds 6, ICO^{FA} starts to

outperform ICO^S noticeably and the drop rate gap between them expands as k grows further.

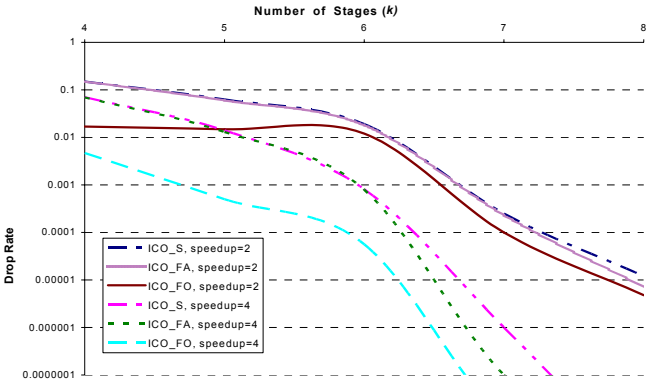


Fig. 7. Packet drop rate versus k for buffered switching fabrics under the load of 1.0.

Comparing these results with those of unbuffered cases reveals that the introduction of buffers to SE's yields considerable performance enhancement for a given k. As an instance, ICO^S with k = 6 delivers a maximum offered load of 0.79 (from Fig. 5) without buffers in SE's, as opposed to more than 0.98 with buffers. Likewise, ICO^{FO} under k = 6 arrives at the maximum offered load of 0.95 without buffers, in contrast to 0.99 with buffers. Alternatively, the switching fabrics with buffers need far fewer stages to achieve a given performance level than their compatible unbuffered ones. For example, the drop rate of our buffered ICO^{FO} (or ICO^S) with k = 6 is smaller than that of its unbuffered counterpart with k = 9 according to Fig. 5, reflecting a significant savings in the total number of SE's involved. While an SE with buffers is more complex than a corresponding SE without buffers, the overall cost of a switching fabric is likely to depend mostly on the number of stages (or SE's), because the number of stages dictates the chip count of a fabric. This reduction in the number of stages also translates to a simpler circuit (i.e., selector) for terminating packets at each destination queue. As a result, a buffered switching fabric appears to be more attractive than its unbuffered counterpart due to its potentially lower cost.

6. Conclusion

This article has introduced scalable switching fabrics based on a multistage structure with different re-circulating designs, referred to as ICO^S, ICO^{FA}, and ICO^{FO}, respectively. With distributed routing, these fabrics aim to interconnect large numbers of line cards (LC's) in high-performance routers. They employ far simpler switching elements (SE's) than an earlier multistage-based switch, known as the closed-loop Shuffleout (CSO), since no distance computing logics are needed. They all outperform a compatible CSO, resulting from re-circulating packets to the last copy of stages where traffic is far lighter. In particular, ICO^{FO} is demonstrated to prevail among all the fabric designs, made possible by employing simple logics to re-circulate packets through best entry points in the last copy (of stages) without wasting resources or causing unnecessary conflicts. It leads to considerable savings in hardware complexity.

When buffers are incorporated in SE's to create a queue for each output port, the number of stages (of SE's) required for a proposed switching fabric to achieve a given performance level is lowered significantly when compared with its unbuffered counterpart, even for a small speedup of 2. This reduction in the number of stages also leads to a simpler logic for terminating packets at each destination queue. As a result, buffered switching fabrics seem to have lower costs than compatible unbuffered ones, because the overall cost of a fabric is determined largely by the stage count (which affects the chip count). Our simulation results reveal that buffered ICO$_N^S$ and ICO$_N^{FA}$ exhibit roughly identical performance for any given k and ζ (speedup), but buffered ICO$_N^{FO}$ clearly prevails, in particular when k is close to $\log_b N$ and ζ is small (say, 2). While buffered ICO^S, ICO^{FA}, and ICO^{FO} all possess good scalability and low overall costs, buffered ICO^{FO} is especially suitable for large sized construction, in particular, under speedup = 2.

References

[1] S. Bassi *et al.*, "Multistage Shuffle Networks with Shortest Path and Deflection Routing for High Performance ATM Switching: The Open-Loop Shuffleout," *IEEE Trans. on Communications*, vol. 42, pp. 2881-2889, Oct. 1994.

[2] S. Bassi *et al.*, "Multistage Shuffle Networks with Shortest Path and Deflection Routing for High Performance ATM Switching: The Closed-Loop Shuffleout," *IEEE Trans. on Communications, vol.* 42, pp. 3034-3044, Nov. 1994.

[3] M. Decina, P. Giacomazzi, and A. Pattavina, "Shuffle Interconnection Networks with Deflection Routing for ATM Switching: the Open-Loop Shuffleout," *Proc. 13th Int'l Teletraffic Conf.*, June 1991, pp. 27-34.

[4] A. L. Gupta and N. D. Georganas, "Analysis of a Packet Switch with Input and Output Buffers and Speed Constraints," *Proceedings of IEEE INFOCOM'91*, Apr. 1991, pp. 694-700.

[5] M. G. Hluchyj and M. J. Karol, "Queueing in High-Performance Packet Switching," *IEEE J. on Selected Areas in Communications,* vol. 6, pp. 1587-1597, Dec. 1988.

[6] M. Katevenis, D. Serpanos, and E. Spyridakis, "Switching Fabrics with Internal Backpressure Using the ATLAS I Single-Chip ATM Switch," *Proceedings of IEEE GLOBECOM'97*, Nov. 1997, pp. 242-246.

[7] N. F. Maxemchuk, "Comparison of Deflection and Store-and-Forward Techniques in the Manhattan Street and Shuffle-Exchange Networks," *Proceedings of IEEE INFOCOM'89*, Apr. 1989, pp. 800-809.

[8] N. McKeown *et al.*, "The Tiny Tera: A Packet Switch Core," *IEEE Micro*, vol. 17, pp. 26-33, Jan./Feb. 1997.

[9] M. Pease, III, "The Indirect Binary *n*-Cube Microprocessor Array," *IEEE Trans. on Computers, vol.* C-26, pp. 250-265, May 1977.

[10] S. Moon, J. Rexford, and K. G. Shin, "Scalable Hardware Priority Queue Architectures for High-Speed Packet Switches," *IEEE Trans. on Computers, vol.* 49, pp. 1215-1227, Nov. 2000.

[11] N. Tzeng, K. Ponnuru, and K. Vibhatavanij, "A Cost-Effective Design for ATM Switching Fabrics," *Proceedings of 1999 IEEE Int'l Conf. on Communications (ICC),* June 1999, pp. S37.4.1-5.

[12] M. Galles, "Spider: A High-Speed Network Interconnect," *IEEE Micro*, vol. 17, pp. 34-39, Jan./Feb. 1997.