Multicast Traffic in Input-Queued Switches: Optimal Scheduling and Maximum Throughput

Marco Ajmone Marsan, *Fellow, IEEE*, Andrea Bianco, *Member, IEEE*, Paolo Giaccone, *Member, IEEE*, Emilio Leonardi, *Member, IEEE*, and Fabio Neri, *Member, IEEE*

***Abstract*—This paper studies input-queued packet switches loaded with both unicast and multicast traffic. The packet switch architecture is assumed to comprise a switching fabric with multicast (and broadcast) capabilities, operating in a synchronous slotted fashion. Fixed-size data units, called cells, are transferred from each switch input to any set of outputs in one time slot, according to the decisions of the switch scheduler, that identifies at each time slot a set of nonconflicting cells, i.e., cells neither coming from the same input, nor directed to the same output.**

First, multicast traffic admissibility conditions are discussed, and a simple counterexample showing intrinsic performance losses of input-queued with respect to output-queued switch architectures is presented. Second, the optimal scheduling discipline to transfer multicast packets from inputs to outputs is defined. This discipline is rather complex, requires a queuing architecture that probably is not implementable, and does not guarantee in-sequence delivery of data. However, from the definition of the optimal multicast scheduling discipline, the formal characterization of the sustainable multicast traffic region naturally follows. Then, several theorems showing intrinsic performance losses of input-queued with respect to output-queued switch architectures are proved. In particular, we prove that, when using per multicast flow FIFO queueing architectures, the internal speedup that guarantees 100% throughput under admissible traffic grows with the number of switch ports.

***Index Terms*—Input queued switches, multicast traffic, scheduling, switching.**

I. Introduction and Previous Work

In THIS paper, we discuss approaches to support multicast traffic in high performance *input-queued cell-based* packet switches. We consider both pure input-queued (IQ) architectures, where buffers reside only at input ports and no internal speedup is required, and combined input and output queued (CIOQ) architectures where buffers reside at both input and output ports, and a moderate internal speedup is provided in the switching fabric. The reason to focus on this class of switch architectures is that they are considered to be the most promising for the implementation of extremely fast packet switches and routers [1], thanks to the fact that the aggregate bandwidth required in their switching fabric does not grow with the sum of the data rates of input links. Instead, output queued (OQ) switch architectures with N input/output ports require switching fabrics and output memories whose bandwidth must be up to N times the sum of the data rates of input links. We consider a *cell-based* paradigm within switches: arriving packets are fragmented into fixed-size cells, which are stored into buffers at input ports. Cells are transferred from input to output ports through the switching fabric, following a scheduling discipline that must avoid contention (no more than one cell for IQ, and k cells for CIOQ with speedup k, can be extracted from an input port in one time slot, and the same constraints apply to the number of cells that can be delivered to an output port in one time slot). Packets are then reassembled at output ports. No limits to buffer capacities are considered.

Several IQ switch architectures have been proposed [2] to solve the problem of transferring a multicast cell in one time slot from an input queue to possibly several output ports. They are based either on internal copy networks or recirculating lines, so that multicast cells are replicated at inputs and treated like unicast cells, or on redundant switching paths, that allow the parallel transfer of multicast cells to their destinations. Note that cell replication at inputs requires some form of memory speedup. From an architectural point of view, the availability within the switch of a switching fabric with *intrinsic multicast capabilities* is extremely important to reduce the cost of multicast traffic management. For example, switching fabrics implemented with a bus or a crossbar offer the possibility of transferring a cell from one input port to many output ports at no extra cost; the cell injected into the switching fabric at the input port can reach any number of output ports within one time slot. We consider in this paper IQ and CIOQ cell-based switches whose internal fabrics have such intrinsic multicast capabilities.

The problem of scheduling multicast traffic in IQ switches was defined and modeled in [3], using a theory based on stochastic ordering and majorization. In that work, the optimal scheduling discipline is fully characterized for a switch with two and three input ports, based on a queueing structure with only one first-in-first-out (FIFO) queue for each input. Larger switches were not considered, and no results about the maximum achievable throughput were provided.

Several theoretical studies [4]–[7] have appeared, that investigate the maximum throughput achievable when arrivals of multicast cells are generated according to a Poisson process, and random services of input queues are assumed. Moreover, cells at the head of input queues are assumed to be served independently across the different inputs, as well as from slot to slot. These models show that the maximum normalized throughput

Manuscript received March 7, 2001; revised June 25, 2002; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor G. Rouskas. This work was supported in part by the Italian Ministry for University and Scientific Research through the MQOS Project. Preliminary versions of this paper were presented at the IEEE INFOCOM 2001, Anchorage, AK, and IEEE International Conference on Communications 2001, Helsinki, Finland.

The authors are with the Dipartimento di Elettronica, Politecnico di Torino, Torino, Italy (e-mail: ajmone@polito.it; bianco@polito.it; giaccone@polito.it; leonardi@polito.it; neri@polito.it).

Digital Object Identifier 10.1109/TNET.2003.813048

for IQ switches under uniform multicast traffic is always less than one, and that it depends on the multicast traffic distribution.

In [8], the multicast scheduling problem was studied in its possible variants (see Section II), and its hardness was proved. Another problem investigated in [8] is the integration of unicast with multicast traffic, suggesting the transfer of unicast cells to their output ports when the multicast schedule leaves those idle, thus, treating multicast traffic as a different class from unicast.

A well-established result in the field of switching is that a CIOQ switch with internal speedup equal to 2 can emulate an OQ architecture [9]. A common belief it is that, by emulating an OQ architecture in a CIOQ switch with speedup equal to 2, it is possible to transfer multicast traffic with no problem. However, this is not true. Unicast traffic has a very "nice" property: to approach saturation on all the N output ports of the switch, it is necessary to receive packets at all the N inputs. Thus, under unicast traffic, the instantaneous aggregate load of the switch is always less than or equal to the total capacity of the switch. On the contrary, with multicast traffic, packets arriving at just one input can bring all the switch outputs close to saturation; this implies that, when all the switch inputs are active, for some time periods the instantaneous aggregate switch load can be N times the total capacity of the switch (consider for example the possibility of sequences of broadcast cells arriving at all inputs); in other words, multicast traffic, even if admissible, can temporarily "flood" the switch, and cannot be scheduled with the approach proposed in [9].

In [10], a speedup equal to 2 was proved to be sufficient to obtain the stability of a CIOQ switch, *provided* that multicast flows satisfy some conditions, corresponding to the fact that the multicast traffic is well regulated, and cannot "flood" the switching fabric. This is a restrictive assumption that cannot be assumed to hold in general.

The main goals of this paper are: 1) to discuss the performance achievable by IQ and CIOQ switch schedulers supporting multicast traffic, under any admissible traffic pattern (i.e., under traffic patterns that overload neither inputs nor outputs) and 2) to define the optimal multicast traffic scheduling algorithm. These results are a starting point to devise low-complexity multicast traffic scheduling algorithms yielding good performance. As a performance metrics we focus on the maximum throughput achievable by an IQ switch, or, equivalently, on the minimum speedup required in a CIOQ switch to achieve the same throughput of an OQ switch. Preliminary results were presented in [11] and [12].

In Section II, we introduce the problem of scheduling multicast traffic in IQ and CIOQ switches, and propose an innovative queueing architecture useful for theoretical considerations. In Section III, the optimal multicast scheduling discipline is defined. Then, in Section IV we identify a class of "worst-case" traffic patterns, i.e., traffic patterns that lead to a minimization of the switch throughput, and, in Section V, we analytically prove that any scheduling algorithm leads to poor performance when IQ and CIOQ switches are loaded with this type of traffic. To ease a first reading of the paper, most analytical derivations and theorem proofs were moved to the Appendixes.

The results in this paper are quite relevant from a theoretical viewpoint, since they prove that IQ and CIOQ architectures are inferior to OQ architectures in the case of general multicast traffic patterns, contrary to the case of unicast traffic, for which IQ and OQ switches were proved to be equivalent [9].

II. Scheduling Discipline and Queueing Architecture

In this section, we introduce some basic definitions about multicast traffic scheduling, and we illustrate the switch architecture considered in this paper.

Unless otherwise specified, we refer to switches with N input and N output ports, where all input and output lines run at the same data rate. The switching fabric is assumed to have intrinsic multicasting (and broadcasting) capabilities, i.e., the cost of transferring in a time slot a cell from one input to one or more outputs does not depend on the number of destinations.

The average amount of traffic at each input (output) is called the input (output) load, and is measured in cells per time slot. We normalize input (output) loads to line rates: a load equal to 1 means a fully utilized input (output) line (one cell per time slot). The traffic at the input of a switch is said to be *admissible* if no input load is larger than 1, and no output load is larger than 1. An input traffic is said to be *sustainable* if it can be transferred through the switch.

Any multicast cell is characterized by its *fanout set*, i.e., by the set of switch output ports (destinations) to which the cell is directed. The cell *fanout* [6], [7] is defined as the number of different destinations of a multicast cell, i.e., the cardinality of the fanout set. We say that a cell has *fanout destination* j when output port j belongs to the fanout set of the cell. A unicast cell has fanout one, and its fanout destination is the only output port to which the cell is destined.

At each time slot, cells stored in input queues contend to access the switching fabric to reach output ports. The decision about which cells can be transferred is made by the switch scheduler, which implements a scheduling discipline. The fact that multicast cells have multiple destinations implies that some scheduling disciplines may elect to transfer in just one time slot the multicast cell to all destinations (in this case we say that no fanout splitting is allowed, and the scheduling discipline is not greedy), while others may elect to transfer the cell in several time slots, reaching nonoverlapping and exhaustive subsets of destinations (in this case fanout splitting is allowed, and the scheduling discipline is greedy). In the latter case, a *partial service* is adopted when a cell reaches a subset of its remaining destinations with its current transfer, whereas a *total service* is adopted when a cell reaches all its remaining destinations with its current transfer. In the case of partial service, the *residue* [6], [13] is defined as the set of fanout destinations that have not yet been reached after a multicast cell is transferred toward output ports. Note that, given the fanout set of multicast cells, the same overall residue cardinality is generated by any greedy scheduling discipline. Each scheduling discipline with partial service uses a specific way of distributing or concentrating the residue among all contending inputs.

In [8], it was proved that the multicast scheduling problem is NP-hard, both with and without fanout splitting.

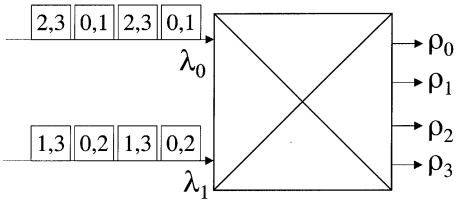

Fig. 1. Traffic pattern leading to poor throughput with both fanout and no fanout splitting, in the case of two active inputs and four active outputs. Numbers inside cells show their fanout destinations.

In the case of fanout splitting, every partial service causes an increase of the input load, leading to performance penalties. Indeed, when fanout splitting is considered, a cell is scheduled in an average number of time slots equal to α, with $\alpha \geq 1$. This fact increases the number of time slots necessary to schedule the cell transfers and thus, either the input load must be lowered by the same factor α (α can be seen as a factor of bandwidth reduction) or a minimum internal speedup equal to α is required. This performance degradation due to excessive splitting, which generates additional load, was observed in [7]. If no fanout splitting is considered, the throughput can drop to very low values, since in this case another form of performance penalty is introduced. Because of the higher throughput achievable, we consider in this paper only scheduling disciplines with fanout splitting.

The example shown in Fig. 1 gives an indication of the problems that can arise with unfortunate but admissible traffic patterns. Inputs are fed with a sequence of back-to-back cells as shown in the figure, with alternating multicast destinations. This traffic is admissible; however, with multicast traffic, the constraints for traffic sustainability are well-defined only for OQ switch architectures: no input load and no output load must be larger than 1. In other words, any admissible traffic is sustainable for an OQ switch. In the case of IQ switches, instead, while the best scheduling discipline for unicast traffic can sustain the same load of an OQ switch, maximum throughput may not be achieved under multicast traffic. Indeed, with the best fanout splitting strategy, $3/2$ cells (i.e., one cell toward both destinations, and one cell toward only one destination, leaving a residue) can be scheduled in each time slot in the whole switch; on average, for each input, $3/4$ cells are scheduled in each time slot. If a no fanout splitting discipline is considered, only one cell can be scheduled in each time slot; on average, for each input, $1/2$ cells are scheduled in each time slot. Thus, whereas an OQ switch can sustain the considered traffic pattern until $\lambda_0 \leq 1$ and $\lambda_1 \leq 1$, where λ_i is the average normalized load at input i, IQ switches must impose $\lambda_0 \leq 0.75$ and $\lambda_1 \leq 0.75$ when a fanout splitting discipline is considered, and $\lambda_0 \leq 0.5$ and $\lambda_1 \leq 0.5$ for no fanout splitting disciplines.

The choice of the queue structure obviously affects the scheduling discipline, since the cells that can be examined in each time slot are always a (small) subset of all cells stored at input ports. On the other hand, the scheduling algorithm is tailored to the chosen queue architecture. Several different ways of organizing the input queue system can be envisaged; recall that, under multicast traffic, buffer space may be used more efficiently by IQ switches than by OQ switches, in the sense that any multicast cell currently in the switch can be stored using only one buffer position.

In an IQ switch, when unicast cells are stored in just one FIFO queue per input port, the cell at the head of the queue can block the access to the switching fabric of subsequent cells, leading to the well-understood head-of-the-line (HoL) blocking effect [14], which limits the maximum throughput achievable by IQ switches. In the case of unicast traffic only, the usual approach to avoid HoL blocking consists of using, at each input, separate queues for each output (thus, N queues per input, and N^2 queues overall); this queueing architecture is called *virtual output queueing* (VOQ).

For multicast traffic, HoL blocking can be completely avoided using at each input separate FIFO queues for each one of the $(2^N - 1)$ possible fanout configurations. We call a *multicast flow* the sequence of cells that arrive at a switch input port with a given fanout set. We shall refer to this queue architecture with the name *multicast virtual output queueing* (MC-VOQ). The (unicast and multicast) scheduling algorithm considers only the cells at the heads of the $2^N - 1$ FIFO input queues: only those cells may be scheduled for transfer to output ports. Hence, the considered scheduling algorithm is an *HoL scheduler*, i.e., it examines only cells at the head of each queue.

Note that our definition of HoL scheduler, although it assumes only FIFO queues, hence, no queue lookahead, does not prevent re-enqueueing a HoL packet to the tail of a different queue. Indeed, in the MC-VOQ architecture, when a multicast cell receives partial service, leaving a residue, we assume that that cell is dequeued from its current queue, and enqueued in the last position of the FIFO queue corresponding to the residue. For example, if a cell with fanout set $\{2, 5, 6\}$ can be transferred only to output 5 in the current time slot, it is dequeued from the queue corresponding to fanout set $\{2, 5, 6\}$ and queued to the FIFO storing packets for fanout set $\{2, 6\}$. The above described approach can lead to out-of-sequence delivery of cells belonging to the same multicast flow.

Although the MC-VOQ architecture entails a very large number of FIFO queues, we show in the next section that it is essential in the definition of the optimal multicast scheduling algorithm which allows us to achieve the maximum possible throughput. The use at each input of a set of FIFO queues is today standard in high-performance switches and routers: the per-flow and per-class queueing architectures are common examples of this type of queue architecture. The MC-VOQ architecture differs from these queue architectures only because of the need for $2^N - 1$ queues at each input; this may impair the switch feasibility, but the complexity of the queueing scheme is the price that has to be paid to completely eliminate HoL blocking.

III. Optimal Scheduling

In this section we define the optimal scheduling discipline for an IQ switch with a MC-VOQ queueing architecture. Our methodology is based on the approach used in [15]–[18]. We first introduce our notation and some useful relations.

A. Switch Description

The queueing architecture is assumed to be MC-VOQ. Let I be the set of input ports, and O the set of output ports. Let N_Q be the total number of queues, $N_Q = N(2^N - 1)$.

We define X_n as the row vector of queue lengths (occupancies) at time $n, n = 0, 1, 2, \ldots$, i.e., $X_n = [X_{n,q_1}, X_{n,q_2}, \ldots, X_{n,q_{N_Q}}]$, being $X_{n,q}$ the length of queue q.

Arrivals at switch inputs are described through a stochastic process A, which is a sequence of arrival vectors $A_n = [a_{n,q_1}, a_{n,q_2}, \ldots, a_{n,q_{N_Q}}]$; $a_{n,q}$ is the number of cells arriving at queue q during time slot n. $D_n = [d_{n,q_1}, d_{n,q_2}, \ldots, d_{n,q_{N_Q}}]$ is the departure vector: $d_{n,q}$ is the number of cells leaving queue q during time slot n.

We assume $a_{n,q} \in \{0, 1\}$ and $d_{n,q} \in \{-1, 0, 1\}$. The -1 value of $d_{n,q}$ is quite unusual, but it is necessary when, due to fanout splitting, only a part of the fanout set of a cell is switched, so that the residue must be buffered at the queue corresponding to the remaining fanout destinations, as discussed above. If a cell is partially transferred from queue q_i, and must be re-enqueued into queue q_j, then $d_{n,q_i} = 1$ and $d_{n,q_j} = -1$.

The queue length evolution is described by the relation

$$X_{n+1} = X_n + A_n - D_n \quad n \geq 0.$$

We assume $X_0 = 0$.

Let w_i^I be a binary row vector of size $N(2^N - 1)$, i.e., $w_i^I = [w_{i,q_1}^I, \ldots, w_{i,q_{N_Q}}^I]$, with $w_{i,q}^I = 1$ iff queue q stores cells from input $i, i = 0, 1, \ldots, N-1$. Similarly to w_i^I, let $w_j^O = [w_{j,q_1}^O, \ldots, w_{j,q_{N_Q}}^O]$, with $w_{j,q}^O = 1$ iff output $j, j = 0, 1, \ldots, N-1$, is included in the fanout set of the cells stored in queue q. Let $w_{ij}^{IO} = w_i^I \wedge w_j^O$, where $\wedge$ denotes logical *and*; the element of the binary vector w_{ij}^{IO} corresponding to queue q is 1 iff q stores packets from input i destined to j. If V is a vector, let $u(V)$ be a vectorial operator which outputs a vector U (with the same size of V) whose kth element u_k is equal to the step function applied to v_k, i.e., u_k is equal to 1 if $v_k > 0$, otherwise $u_k = 0$.

The technological constraints in the switching fabric are described by the following linear inequalities:

$$0 \leq w_i^I A_n^T \leq 1 \tag{1}$$
$$0 \leq w_i^I u\left(D_n^T\right) \leq 1 \tag{2}$$
$$0 \leq w_i^I u\left(-D_n^T\right) \leq 1 \tag{3}$$
$$0 \leq w_j^O u\left(D_n^T\right) \leq 1 \tag{4}$$
$$0 \leq w_{ij}^{IO} D_n^T \leq 1 \tag{5}$$

for $i \in I, j \in O$ and $n \geq 0$. Equation (1) means that at most one cell can arrive at each input during a time slot, while (2) means that at most one cell can be forwarded from each input. At most one cell can be re-enqueued into a queue, according to (3); this implies a memory access speedup equal to 2: at a given queue in each time slot at most one cell arrives and one cell is re-enqueued. Equation (4) means that at most one cell is sent to an output at each time slot; by (5), each cell which receives a partial service is moved to a queue corresponding to a fanout set which is a subset of the initial one. Indeed, (5) states that three cases are possible:

TABLE I
EXAMPLE OF MC-VOQ QUEUEING SYSTEM IN A SWITCH WITH $N = 2$

Queue	Input	Fanout set
q_0	0	0
q_1	0	1
q_2	0	0,1
q_3	1	0
q_4	1	1
q_5	1	0,1

1) no multicast cell with destination j is transferred from input i, so that $w_{ij}^{IO} D_n^T = 0$;
2) one multicast cell is transferred from input i to output j, so that $w_{ij}^{IO} D_n^T = 1$; the cell transfer either reaches all destinations, or leaves a residue, but the latter does comprise output j;
3) one multicast cell is transferred from input i to a subset of its destinations, which does not include output j, so that the cell must be re-enqueued at the input queue corresponding to its residue, and $w_{ij}^{IO} D_n^T = 1 - 1 = 0$. By looking at all outputs of this cell, we see that re-enqueueing takes place only at a queue referring to outputs that were not reached, hence, at a queue corresponding to a fanout set which is a subset of the initial one.

For example, in the case $N = 2$, with $I = O = \{0, 1\}$ and the fanout set of the queues described by Table I, we have $w_0^I = [1\,1\,1\,0\,0\,0]$, $w_1^I = [0\,0\,0\,1\,1\,1]$, $w_0^O = [1\,0\,1\,1\,0\,1]$, $w_1^O = [0\,1\,1\,0\,1\,1]$; $w_{00}^{IO} = [1\,0\,1\,0\,0\,0]$, $w_{01}^{IO} = [0\,1\,1\,0\,0\,0]$, $w_{10}^{IO} = [0\,0\,0\,1\,0\,1]$, $w_{11}^{IO} = [0\,0\,0\,0\,1\,1]$.

B. Admissible Region

The arrival vector at time n, A_n, is a realization of a stochastic process characterized by the row vector Λ, whose elements are the average arrival rates at queues: $\Lambda \triangleq E[A_n]$.

We define the input traffic to be admissible when neither input nor output ports are overloaded. With the notation that we introduced before, input ports are not overloaded if

$$0 \leq w_i^I \Lambda^T \leq 1 \quad \forall i \in I. \tag{6}$$

In a similar way, output ports are not overloaded if

$$0 \leq w_j^O \Lambda^T \leq 1 \quad \forall j \in O. \tag{7}$$

Note that (6) and (7) are necessary but not sufficient conditions for the rate stability of an IQ switch, while they guarantee the rate stability of an OQ switch.

The set of vectors Λ satisfying (6) and (7) forms the *admissible region* $\mathcal{A}$ for input traffic.

C. Throughput Definition

The switch throughput is usually measured at output ports. The instantaneous throughput $\rho_n(j)$ at output port j at time n, and the average throughput $\rho(j)$ at output port j are defined as follows:

$$\rho_n(j) \triangleq w_j^O u\left(D_n^T\right) \quad \rho(j) \triangleq E[\rho_n(j)].$$

D. Optimal Scheduling and Capacity Region

The scheduling of multicast traffic in an IQ switch can be formalized as a convex analysis problem.

Referring to the stochastic version of Lyapunov stability [15], the maximum throughput of the switch can be obtained, as proved in Appendix A, by solving the following optimization problem at each time slot n:

$$D_n^\star = \arg\left\{ \max_{D_n} \{D_n X_n^T\} \right\} \tag{8}$$

subject to constraints (2), (3), (4), and (5). $D_n^\star$ defines the optimal scheduling choice at time slot n. We call *max-scalar discipline* the scheduling algorithm that selects $D_n^\star$ at each time slot n. The proof of the optimality of the max-scalar discipline is reported in Appendix A.

As shown in Appendix A, a traffic pattern is sustainable if the average arrival rate vector Λ lays within the convex hull generated by all possible departure vectors $D^{(k)}$, i.e., the arrival rate vector Λ must be such that

$$\Lambda = \sum_k \alpha_k D^{(k)} \tag{9}$$

$$\text{subject to} \quad \begin{cases} \sum_k \alpha_k = 1 \\ 0 \le \alpha_k \le 1 \quad \forall k. \end{cases} \tag{10}$$

The set of all arrival rate vectors Λ that satisfy the above expressions defines the *capacity region* $\mathcal{C}$ for multicast traffic in an IQ switch. The capacity region contains all arrival rate vectors for which the queues of the IQ switch remain finite. The examples and the theorems in Section V prove that the capacity region is strictly internal to the admissible region. It is worth observing that for OQ switches the two regions coincide.

The implementation of the max-scalar scheduling discipline requires solving a linear programming problem at each time slot, but a numerical solution is extremely complex because of the huge number of possible departure vectors $D^{(k)}$, which corresponds to the number of possible switching configurations. In Appendix B this number is computed, and results are reported in Table II. Note the very fast growth of the problem complexity.

Starting from the definition of the admissible region $\mathcal{A}$ and of the capacity region $\mathcal{C}$, we can formally define the minimum speedup to sustain all admissible traffic patterns in an IQ switch, as follows:

$$S_{\text{MIN}} = \arg\left\{ \inf_{1 \le s \le N} \left\{ \frac{1}{s} \Lambda \in \mathcal{C}, \forall \Lambda \in \mathcal{A}, \forall N \right\} \right\}. \tag{11}$$

Unfortunately, the numerical evaluation of (11) is prohibitive. We will later show that S_{MIN} is not bounded when the switch size grows to infinity.

IV. K-Complex Traffic Definition

In this section, we describe a multicast traffic pattern which proves to be critical for IQ switches.

We introduce the following definitions:

- $O = \{o_k\}_{k=0}^{N-1}$ is the set of switch output ports; $|O| = N_o = N$;
- $I = \{i_k\}_{k=0}^{N-1}$ is the set of switch input ports; $|I| = N_i = N$;

TABLE II
NUMBER OF POSSIBLE DEPARTURE VECTORS IN AN IQ MULTICAST SWITCH

Number of ports	Number of switching configurations
2	19
4	$1.95\ 10^5$
8	$3.42\ 10^{25}$
16	$3.71\ 10^{90}$

- $A \subseteq I$ is the set of *active* switch input ports, i.e., input ports with cells waiting to be switched; $|A| = N_a$;
- $R = \{r_l\}$ is the *request set* of (multicast) cells waiting to be switched at input queues;
- $D_l = \{d_{lk}\}$ is the set of destinations of the lth cell waiting to be switched;
- $D_{lm} \subseteq D_l$ is the set of destinations of the lth cell that are reached with the mth transfer of the lth cell through the switch (in the case of fanout splitting); we assume to split the fanout set in disjoint subsets: $\cup_m D_{lm} = D_l$, and $D_{lm} \cap D_{ln} = \emptyset, \forall m \ne n$.

We now describe a particular class of multicast traffic patterns, which is essential for our results.

Definition 1: A request set R is said ***k*-complex**, with[1] $k \in \mathbb{N}, k > 1$, if

1) k cells are queued at each active input $a_l \in A$;
2) k cells are directed to each output $o_l \in O$;
3) for each subset of R comprising k cells, a destination exists to which all the cells in the subset are directed.

Table III reports an example of a 2-complex request set for a 6×6 switch, where only inputs i_0 and i_1 are active.

In an $N \times N$ switch where N_a input ports are active, with $N = \binom{kN_a}{k}$ and $N_a \ge 2$, a k-complex request set R of size kN_a can be generated with the following algorithm:

Step 1) Assign the first k cells in R to the first input, the second k cells to the second input, and so on, until the last k cells have been assigned to input N_a. With reference to Table III, $R = \{R_0, R_1, R_2, R_3\}$, and $\{R_0, R_1\} \Leftrightarrow i_0$ and $\{R_2, R_3\} \Leftrightarrow i_1$.

Step 2) Form all the possible $N = \binom{kN_a}{k}$ different subsets of R whose size is k, and create an arbitrary injective correspondence from subsets of cells to destinations. With reference to Table III, $\{R_0, R_1\} \Leftrightarrow o_0$, $\{R_0, R_2\} \Leftrightarrow o_1$, $\{R_0, R_3\} \Leftrightarrow o_2$, $\{R_1, R_2\} \Leftrightarrow o_3$, $\{R_1, R_3\} \Leftrightarrow o_4$, and $\{R_2, R_3\} \Leftrightarrow o_5$.

Step 3) To each cell in R assign all the destinations that correspond to sets containing the cell itself. With reference to Table III, $R_0 = \{o_0, o_1, o_2\}, R_1 = \{o_0, o_3, o_4\}, R_2 = \{o_1, o_3, o_5\}$, and $R_3 = \{o_2, o_4, o_5\}$.

For each cell, the resulting fanout is equal to $\binom{kN_a - 1}{k-1} = (N/(N_a))$.

Definition 2: A request set R' is called **generalized-*k*-complex**, with $k \in \mathbb{N}, k > 1$, if it contains kN_a cells independently extracted, possibly with repetitions, from a k-complex request set.

[1]In this paper, $\mathbb{N}$ denotes the set of non negative integers, $\mathbb{R}$ denotes the set of real numbers, and $\mathbb{R}^+$ denotes the set of nonnegative real numbers.

TABLE III
EXAMPLE OF A 2-COMPLEX REQUEST SET FOR A 6×6 SWITCH WITH TWO ACTIVE INPUT PORTS

Input	Fanout sets	
i_0	$\{o_0, o_1, o_2\}$	$\{o_0, o_3, o_4\}$
i_1	$\{o_1, o_3, o_5\}$	$\{o_2, o_4, o_5\}$

Note that a generalized-k-complex request set satisfies Condition 3 of Definition 1, but does not necessarily satisfy Conditions 1 and 2, since the number of cells queued at each active input and directed to each output can be different from k. Note also that the conflicts in a generalized-k-complex request set are not less than the conflicts in the corresponding k-complex request set.

Given a request set, we can define two types of traffic patterns:

1) **stochastic** traffic patterns are obtained by offering, at switch inputs, cells extracted at random according to a uniform distribution from the request set at a rate dependent on the desired load;
2) **persistent** traffic patterns are obtained by periodically offering at switch inputs all the cells of the request set; the order of cells must meet the following constraints:
   - at most one cell is offered at any input in a time slot;
   - all the cells in the request set are offered at inputs in the minimum number of time slots, i.e., in a number of time slots equal to the maximum number of cells that arrive at any input or are destined to any output.

For example, if R is a k-complex request set, the traffic pattern obtained by repeating R at switch inputs every k time slots is called a persistent k-complex traffic pattern.

It is important to observe that a persistent k-complex traffic pattern implies a load of input and output ports equal to 1.

V. MAIN RESULTS

In this section, we present our original results, first on the sustainable region of an IQ switch with multicast traffic, then on the minimum speedup necessary to schedule a k-complex traffic pattern.

Theorem 1: There exist admissible input multicast traffic patterns that lead to 100% throughput in OQ switches, under which IQ switches using a no fanout splitting policy provide a throughput that can drop to zero if the number of ports grows to infinity.

Proof: We consider an (OQ or IQ) switch with N_a active input ports receiving a persistent 2-complex multicast traffic pattern.

An OQ switch architecture can transfer kN_a packets every $k = 2$ time slots, i.e., an average of N_a packets per time slot, and can thus sustain any admissible load; this is the case for the persistent traffic pattern that we are considering.

On the contrary, in an IQ switch using a no fanout splitting policy, under the same persistent 2-complex traffic pattern, at most one cell can be scheduled in each time slot. Thus, the throughput for IQ switches with no fanout splitting is $1/N_a$ of the throughput for OQ switches, and thus, drops to zero when N_a grows indefinitely. □

Theorem 2: There exist admissible input multicast traffic patterns, under which the maximum sustainable throughput for an IQ switch using a fanout splitting policy is not greater than 0.5.

Proof: To prove the theorem, we present a case in which the maximum load per input should be less than or equal to 0.5 for the IQ switch to be stable.

We consider a large size IQ switch loaded with a stochastic k-complex traffic pattern, in which only k input ports are active, and the offered load, measured as the average number of cells arriving at each input during each time slot, is λ. The effective service rate of each input queue, measured as the average number of packets transferred from each queue, is μ. It is necessary that $\lambda < \mu$ to guarantee the system stability, in the sense that all queue occupations are kept finite.

In each time slot, at most one multicast packet can be completely transferred, due to the properties of k-complex traffic (the complete transfer of two packets in the same time slot would lead to a conflict on the output port to which both packets are directed). As a consequence, the fanout of most packets must be split in at least two parts by any greedy scheduler. Thus, at each time slot, at most one packet can be completely transferred, and no more than $k-1$ packets can be partially transferred. Hence, the maximum number of packets transferred in each time slot cannot exceed

$$T = 1 + \frac{k-1}{2}$$

where the partial transfer of a packet whose fanout is split in two counts as half packet transfer, since two time slots are required to completely transfer the packet. The effective service rate per input port can then be written as

$$\mu = T/k = \frac{1}{k} + \frac{k-1}{2k} = \frac{k+1}{2k}.$$

Since $\mu > 0.5$ and $\lim_{k\to\infty} \mu = 0.5^+$, the stability condition requires that $\lambda \leq 0.5$. □

Note that Theorem 2 can be referred also to switching architectures with internal speedup, i.e., to CIOQ architectures: the minimum speedup required to achieve 100% throughput under any admissible traffic pattern in CIOQ switches is not less than two for large switch size.

We now come to the determination of the minimum speedup necessary to schedule a k-complex traffic pattern. Consider CIOQ switches, which provide an internal speedup, and first focus on the minimum number of slots required to transfer from the inputs to the outputs of a switch all the cells belonging to a request set. We call *time frame* a set of contiguous time slots. Theorem 8 of Appendix C shows that, given a k-complex request set R with $|R| = kN_a$, for any finite integer $S \geq 2$, there exists an integer N_0 such that, $\forall N_a > N_0$, it is not possible to schedule all the cells in R with a frame length smaller than Sk.

This important result can be immediately applied to the general class of *frame-based* schedulers, but will also be later used to show properties of slot-by-slot schedulers. In switches adopting a frame-based scheduler, a fixed number of time slots is grouped into a frame of fixed length, say, L time slots, at both inputs and outputs. Assuming that the switch operates

with speedup S (we assume S to be an integer, for the sake of simplicity), the scheduler works on an internal frame of length SL time slots (note however that, due to the internal speedup, the internal frame duration in time units coincides with the input and output frame duration). The scheduler allocates time slots of the internal frame for the transfer of multicast cells considering the state of input queues at the beginning of the frame. The scheduling constraints are that no more than one cell can be scheduled from any input and to any output during each time slot of the internal frame. The scheduling problem is in this case called *time slot assignment*, and it is formally defined in Appendix C. Frame-based schedulers can have interesting applications, since they may foster a simpler solution than slot-by-slot schedulers to scheduling traffic with strict quality-of-service (QoS) guarantees in IQ switches. It can be shown that the set of all frame-based schedulers is equivalent to the set of all slot-by-slot schedulers, provided that slot-by-slot schedulers introduce an additional delay equal to the frame length.

The following theorem is a rephrasing of Theorem 8 in Appendix C.

Theorem 3: Consider a switch with N_a active inputs loaded by a persistent (or stochastic at load 1) k-complex traffic pattern, such that $|R| = kN_a$. For any finite S, there exists an integer N_0 such that $\forall N_a > N_0$, no frame-based scheduler can transfer all the cells at the input queues within one internal frame of size Sk.

Theorem 3 is not a proof that a frame-based scheduler cannot achieve 100% throughput under a persistent k-complex traffic pattern in the case of finite frame size: the switch operates on a sequence of frames, and one may conjecture that a request that cannot be accommodated in a given frame can be transferred in subsequent frames at no extra cost, i.e., without increasing the duration of these subsequent frames. We conjecture that this is not the case, i.e., that 100% throughput cannot be achieved under a k-complex traffic pattern, but we were unable to obtain a formal proof for general schedulers. However, when k grows to infinity, Theorem 3 indeed proves that a frame-based scheduler cannot achieve 100% throughput under a k-complex traffic pattern. Moreover, in Theorem 4 we provide a proof of the above throughput limitation for a particular class of slot-by-slot schedulers.

Since the class of frame-based schedulers is equivalent to the class of slot-by-slot schedulers (except possibly for delays), Theorem 3 also implies that IQ and CIOQ switches implementing a slot-by-slot HoL scheduling algorithm, when k grows to infinity, and, as a consequence, also the switch size grows to infinity, cannot achieve 100% throughput.

For finite size switches, Theorem 3 applies directly to periodic (CBR) traffic with bounded delay requirements: there exists a persistent traffic pattern that cannot be scheduled in large IQ and CIOQ switches with any finite speedup, meeting the delay constraints. This means that the deployment of large IQ or CIOQ switches can lead to performance degradations for real-time applications with QoS requirements.

Considering that a generalized-k-complex request set does not reduce conflicts between cells with respect to the corresponding k-complex request set, the following Corollary immediately follows from Theorem 3:

Corollary 1: Consider a switch with N_a active inputs where a persistent generalized-k-complex traffic pattern is enqueued, such that $|R'| = kN_a$. For any finite S, there exists an integer N_0 such that $\forall N_a > N_0$ it is not possible to schedule all the cells at input queues within an internal frame of size Sk.

We now go back to slot-by-slot HoL schedulers. Remember that we call multicast flow the sequence of cells that arrive at one input port of the switch with identical fanout set. A *HoL flow-blocking scheduler* is a HoL slot-by-slot scheduler which does not allow interleaving of cells belonging to the same multicast flow; i.e., the (possibly partial) transfer of the ith cell of flow f is enabled only after the transfer of all preceding $i-1$ cells of the same flow f has been completed. This solution seems quite reasonable if in-sequence delivery of data must be guaranteed. Note that a FIFO HoL flow-blocking scheduler prevents cell re-enqueueing, which was instead assumed in our MC-VOQ architecture of Section II.

We can now state our main result about slot-by-slot scheduling of multicast traffic.

Theorem 4: Consider a switch with N_a active inputs loaded by a persistent k-complex traffic pattern, such that $|R| = kN_a$. For any finite speedup value S, there exists an integer N_0 such that $\forall N_a > N_0$, no HoL flow-blocking scheduler achieves 100% throughput.

Proof: The theorem can be proved by contradiction; suppose that there exists a HoL flow-blocking scheduler that achieves 100% throughput with speedup S under a persistent k-complex traffic pattern. This implies that, on average, the transfer of $2kN_a$ cells is completed within two internal frames of total length $2kS$ time slots. Then, there must exist a pair of internal frames of length $2kS$ in which the transfer of at least $2kN_a$ cells is completed (see Lemma 4 in Appendix C). Since the scheduler is HoL flow-blocking, no more than kN_a cells may be simultaneously handled by the scheduler; thus, of the $2kN_a$ cells whose transfer is completed within a pair of internal frames of length $2kS$, at least kN_a have been completely transferred within the considered pair of internal frames (the first kN_a completions may refer to cells whose transfer may have started before the beginning of the considered pair of frames). However, since any set of kN_a cells belonging to a k-complex request set forms a generalized-k-complex request set, for Corollary 1, they cannot be completely scheduled in a frame whose length is less or equal to Sk. □

While Theorem 3 provided only an asymptotic indication of the impossibility for slot-by-slot schedulers of achieving 100% throughput, Theorem 4 confirms this limitation for finite size switches on a restricted class of slot-by-slot schedulers. Note however that the restriction of a FIFO service of multicast cells belonging to the same flow seems quite reasonable for a wide range of applications, and in particular for real-time multimedia traffic streams, such as video, sound, and voice.

VI. Conclusion

In this paper, we presented a formal description of the multicast traffic scheduling problem in input-queued packet switches with internal multicast capabilities.

Traffic admissibility conditions were defined, and some theorems were proved to point out intrinsic performance losses of input-queued switches with respect to output-queued architectures.

An optimal multicast scheduling discipline, called max-scalar, for the transfer of multicast packets through the switch was defined, and a proof of its optimality, based upon the use of Lyapunov functions, was provided. From the definition of the optimal scheduling discipline, the formal characterization of the capacity region, i.e., of the set of input loads that an ideal IQ switch is able to transfer to output ports, was obtained. In particular, we proved that for large-size switches, when using per-flow FIFO queueing architectures, no finite speedup guarantees 100% throughput under admissible multicast traffic patterns.

The results in this paper are quite relevant from a theoretical viewpoint, since they prove that IQ and CIOQ architectures are inferior to OQ architectures in the case of general multicast traffic patterns, contrary to the case of unicast traffic, for which IQ and OQ switches were proved to be equivalent.

APPENDIX A
OPTIMALITY OF THE MAX-SCALAR POLICY

In the following definitions, we indicate with $\|Y\|$ the Euclidean norm of vector Y, with $\Pr\{H\}$ the probability of event H, and with $E[X]$ the expected value of random variable X.

Definition 3: A system of queues achieves 100% *throughput*, or is *rate stable*, if

$$\lim_{n\to\infty}\frac{X_n}{n}=\lim_{n\to\infty}\frac{1}{n}\sum_{i=0}^{n-1}(A_i-D_i)=0 \quad \text{with probability 1.}$$

Definition 4: A system of queues is said to be *weakly stable* if, for every $\epsilon>0$, there exists $B>0$ such that

$$\lim_{n\to\infty}\Pr\{\|X_n\|>B\}<\epsilon.$$

Definition 5: A system of queues is said to be *strongly stable* if

$$\lim_{n\to\infty}\sup E[\|X_n\|]<\infty.$$

Theorem 5: If there exists a symmetric positive definite matrix $W\in\mathbb{R}^{N\times N}$, and two positive real numbers $\epsilon\in\mathbb{R}^{+}$ and $B\in\mathbb{R}^{+}$, such that, given the function $V(X_n)=X_nWX_n^T$ (called Lyapunov function), it holds

$$E[V(X_{n+1})-V(X_n)\,|\,X_n]<-\epsilon\|X_n\|$$

and

$$E[V(X_{n+1})\,|\,X_n]<\infty$$

for every X_n such that $\|X_n\|>B$, then the system of queues is strongly stable. In addition, all the polynomial moments of the queue length distribution are finite.

The proof is reported in [16] and [18].

Definition 6: Let $CH[\{D^{(k)}\}_k]$ represent the *convex hull* generated by all possible admissible departure vectors $D^{(k)}$.

Theorem 6: A necessary condition to achieve 100% throughput in an IQ switch is that the input traffic pattern, described by a sequence of random arrival vectors A_n and satisfying the strong law of large numbers, hence, stationary and ergodic, satisfies the following condition:

$$E[A_n]\in CH\left[\left\{D^{(k)}\right\}_k\right].$$

Proof: Suppose that the switch achieves 100% throughput, i.e.

$$\lim_{n\to\infty}\frac{X_n}{n}=0 \quad \text{with probability 1}$$

i.e., the length of the queues remains finite, or at most grows to infinity with sublinear rate. As a consequence:

$$\begin{aligned}\lim_{n\to\infty}\frac{X_n}{n}&=\lim_{n\to\infty}\frac{\sum_{i=0}^{n-1}(A_i-D_i)}{n}\\&=\lim_{n\to\infty}\left[\frac{\sum_{i=0}^{n-1}A_i}{n}-\frac{\sum_{i=0}^{n-1}D_i}{n}\right]\\&=E[A]-\lim_{n\to\infty}\frac{\sum_{i=0}^{n-1}D_i}{n}=0\end{aligned}$$

with probability 1.

A is the stationary and ergodic arrival process, and A_n the random arrival vector at time n; due to stationarity, $E[A]=E[A_n]$ holds.

The above equality implies that there exists a sequence of departure vectors such that

$$E[A]=\lim_{n\to\infty}\frac{\sum_{i=0}^{n-1}D_i}{n} \quad \text{with probability 1.}$$

This means that, $\forall\epsilon>0$, with probability 1, there exists a n_0 such that for all $n>n_0$

$$\left\|E[A]-\frac{\sum_{i=0}^{n-1}D_i}{n}\right\|<\epsilon.$$

Note that $(\sum_{i=0}^{n-1}D_i/n)$, for each n, is in $CH[\{D^{(k)}\}_k]$ by definition. As a consequence, $E[A]$ is the limit of a sequence of vectors in $CH[\{D^{(k)}\}_k]$, and thus, it is in $CH[\{D^{(k)}\}_k]$, since $CH[\{D^{(k)}\}_k]$ is a closed set. □

Theorem 7: An IQ switch implementing the max-scalar discipline is strongly stable for each traffic pattern such that $E[A_n]$ is an internal point of $CH[\{D^{(k)}\}_k]$.

Proof: The assertion can be proved by using the Lyapunov function methodology. Since $E[A_n]\in CH[\{D^{(k)}\}_k]$ it immediately follows that, for all $X_n\neq 0$

$$\frac{E[A_n]X_n^T}{\|X_n\|}<\frac{\hat{A}_nX_n^T}{\|X_n\|}=\frac{E[A_n]X_n^T}{\|X_n\|}\alpha\le\frac{\max_{D_n}D_nX_n^T}{\|X_n\|}$$

where $\hat{A}_n$ is the vector parallel to $E[A_n]$ on the border of $CH[\{D^{(k)}\}_k]$, and $\alpha=\|\hat{A}_n\|/\|E[A_n]\|>1$. Denoting $\arg\{\max_{D_n}\{D_nX_n^T\}\}$ by $D_n^{\star}$ as in (8), we can thus write

$$\frac{(\hat{A}_n-D_n^{\star})X_n^T}{\|X_n\|}\le 0. \tag{12}$$

Since $\hat{A}_n=\alpha E[A_n]=E[A_n]-(1-\alpha)E[A_n]$, we can substitute in (12) and write

$$\frac{(E[A_n]-(1-\alpha)E[A_n]-D_n^{\star})X_n^T}{\|X_n\|}\le 0.$$

Hence, if $r_{\min}$ is the minimum nonnull component of vector $E[A_n]$

$$\lim_{\|X_n\|\to\infty} \frac{(E[A_n]-D_n^\star)X_n^T}{\|X_n\|} \le (1-\alpha)\frac{E[A_n]X_n^T}{\|X_n\|} \le (1-\alpha)r_{\min}\frac{\|X_n\|_1}{\|X_n\|} \le (1-\alpha)r_{\min} < 0$$

where $\|X\|_1$ is the sum of the elements of X, which can be shown to be always larger than $\|X\|$. Note that $(1-\alpha)$ is a negative nonnull quantity. By using the quadratic Lyapunov function associated with the identity matrix (see [15] and [17]), i.e., $V(X_n) = X_n X_n^T$, and assuming the max-scalar scheduling discipline, we can write

$$\begin{aligned} &E[V(X_{n+1}) - V(X_n) \,|\, X_n] \\ &\quad = E\left[X_{n+1}X_{n+1}^T - X_nX_n^T \,|\, X_n\right] \\ &\quad = E\left[(X_n + A_n - D_n^\star)(X_n + A_n - D_n^\star)^T - X_nX_n^T\right] \\ &\quad = E\left[2(A_n - D_n^\star)X_n^T + (A_n - D_n^\star)(A_n - D_n^\star)^T\right]. \end{aligned}$$

Note that $\|A_n\| \le N_Q$ and $\|D_n\| \le N_Q$. Hence, $(A_n - D_n^\star)(A_n - D_n^\star)^T$ (which is $\le 2N_Q$) is negligible with respect to $2(A_n - D_n^\star)X_n^T$ for $\|X_n\| \to \infty$; if we choose $\epsilon = -(1-\alpha)r_{\min}/2$, we observe that there exists $B > 0$ and $\epsilon > 0$ such that, for each $X_n : \|X_n\| > B$

$$E[V(X_{n+1}) - V(X_n) \,|\, X_n] < -\epsilon\|X_n\|.$$

If in addition $E[A_nA_n^T] < \infty$, then $E[V(X_{n+1}) \,|\, X_n] < \infty$ $\forall X_n$; since the conditions of Theorem 5 hold, the system is strongly stable. □

Since the strong stability of a system also implies 100% throughput, by combining the previous results, Corollary 2 follows.

Corollary 2: Under any i.i.d. sequence A_n of arrivals such that $E[A_nA_n^T] < \infty$, the max-scalar scheduling discipline maximizes the stability region of the switch, i.e., the switch cannot be stable under any other scheduling discipline if it is unstable under the max-scalar scheduling discipline.

Appendix B
Multicast Departure Vectors

Let us define $F(i,j,k)$ as the number of possible departure vectors in a IQ switch with i inputs, j total outputs and k outputs still available, i.e., outputs that have not been fully scheduled yet.

The number of possible $D^{(k)}$ in an $L \times N$ switch is given by $F(L,N,N)$, which can be computed with the following recursive formula:

$$F(i,j,k) = F(i-1,j,k) + \sum_{s=1}^{k} \binom{k}{s} 2^{j-s} F(i-1,j,k-s) \tag{13}$$

when $1 < i,j \le N$ and $1 \le k \le j$. Furthermore, $F(i,j,k) = 1$ when $i = 0$ or $j = 0$ or $k = 0$, since we include the all-zeros departure vector.

Let us consider (13). $F(i-1,j,k)$ is the number of permutation matrices when the ith input is not scheduled. The fanout of the considered permutation for the ith input is s. The binomial factor is the number of fanout sets which can be generated by s outputs included in the k outputs still available. 2^{j-s} is the number of possible output configurations that can include the considered fanout set (of s elements). With a fanout set of s elements, $(i-1)$ remaining inputs, j total outputs and $(k-s)$ outputs not yet scheduled, $\binom{k}{s}2^{j-s}F(i-1,j,k-s)$ is the total number of permutation matrices that have to be added to the permutation matrix in which the ith input is not scheduled.

Appendix C
On the Minimum Frame Lengths for Complete Multicast Schedules

We first modify and extend the definitions introduced at the beginning of Section IV in the following way, referring to scheduling a request set in a time frame.

- $R = \{r_k\}$ is the request set of (multicast) cells waiting to be switched at input queues at the beginning of a time frame.
- $F = \{f_k\}$ is the set of time slots in a switching frame; $|F| = L$; L is the frame length, whose minimum value necessary for the transfer of all cells in R will be discussed in this appendix; note that at most NL cells can be transferred in a frame of length L.

Each (multicast) cell $r_l \in R$ is associated with a pair (i_l, D_l), where $i_l \in I$ is the input port, and $D_l \subseteq O$ is the set of destination output ports.

Definition 7: Two multicast cells r_l and r_k are said to be in conflict if they either are associated with the same input port ($i_l = i_k$), or have at least one destination in common ($D_l \cap D_k \neq \emptyset$).

In the switch architecture we consider, a subset of the cells in R can be transferred from input ports to output ports in the same time slot, provided that no cells in the subset are in conflict. Conflicts may force the transfer of a cell r_l to disjoint subsets of D_l in different time slots, since we accept fanout splitting.

Definition 8: A scheduling $\mathcal{S}[R]$ is defined as a function whose domain is R and whose image is[2] $U = 2^{I\times 2^O \times F}$. U is the set of transfer units, whose elements u_{lk} associate with each multicast cell $r_l \in R$ a triple (i_l, D_{lk}, f_k), where $i_l \in I$ is the cell input port, $D_{lk} \subseteq D_l$ is a nonempty subset of the cell destinations, and $f_k \in F$ is a time slot in the frame.

Informally, $\mathcal{S}$ assigns to each multicast cell $r_l \in R$ a set of transfer units $u_{lk} = (i_l, D_{lk}, f_k) \in U$; each transfer unit allows a (possibly partial) transfer of the multicast cell, i.e., a transfer toward a subset of the cell destinations.

Definition 9: A scheduling $\mathcal{S}[R]$ is *admissible* if any two elements $(i_1, D_{1k}, f_k) \in U$ and $(i_2, D_{2n}, f_n) \in U$, such that $f_k = f_n$, are nonconflicting, i.e.

$$i_1 \neq i_2$$

and

$$D_{1k} \cap D_{2n} = \emptyset.$$

[2]Given a set A, 2^A denotes the power set of A.

In other words, no (possibly partial) transfer of two different multicast cells can occur in the same time slot of the frame if conflicts arise.

Definition 10: An admissible scheduling $\mathcal{AS}[R]$ is *complete* if $\cup_k D_{lk} = D_l, \forall r_l \in R$, i.e., if it achieves a complete transfer of all cells belonging to R, using a subset of time slots belonging to F.

Definition 11: A time slot assignment $\mathcal{TSA}[R]$ is defined as a function whose domain is R and whose image is the power set of F ($\mathcal{TSA}[R] : R \rightarrow 2^F$).

Informally, a $\mathcal{TSA}$ defines a correspondence between each cell $r_l \in R$ and the set of time slots in the frame in which copies of the cell are transferred.

A more complex, but more useful, definition of a $\mathcal{TSA}[R]$ will allow us to relate a time slot assignment $\mathcal{TSA}[R]$ to an admissible scheduling $\mathcal{AS}[R]$.

Definition 12: A timing function $\mathcal{T}[U] : U \rightarrow 2^F$ is a function that, given a set $\{(i_l, D_{lk}, f_k)\}_k$, returns a set containing all the third components f_k of each element.

Informally, the timing function $\mathcal{T}$ extracts the time slot information related to (possibly partial) multicast cell transfers.

Definition 13: A time slot assignment $\mathcal{TSA}[R] = \mathcal{T}(\mathcal{AS}[R])$ is a composition of two functions, a timing function $\mathcal{T}$ applied to the image of an admissible scheduling function $\mathcal{AS}$.

Although a time slot assignment $\mathcal{TSA}[R]$ does not completely specify the scheduling of multicast cells to be transferred from sources to destinations (it only defines the set of used time slots, with no information about sources and destinations), we will concentrate our attention only on $\mathcal{TSA}[R]$.

Definition 14: A time slot assignment $\mathcal{TSA}[R]$ is said to be *complete* if there exists a complete admissible scheduling $\mathcal{AS}$ such that $\mathcal{TSA}[R] = \mathcal{T}(\mathcal{AS}[R])$.

Definition 15: For each subset $A \subseteq R$, let $\mathcal{I}_{\mathcal{TSA}}(A)$ be the image of A through $\mathcal{TSA}[R]$, i.e., the set comprising all time slots in which some element of A is transferred.

Lemma 1: Let R be a k-complex request set. Consider a complete $\mathcal{TSA}[R]$ and any set $D \subseteq R$ such that $|D| = k$; then, $|\mathcal{I}_{\mathcal{TSA}}(D)| \geq k$.

Proof: Suppose that there exists a subset $D \subseteq R$, with $|D| = k$, such that $|\mathcal{I}_{\mathcal{TSA}}(D)| < k$; since the request set is k-complex, there exists a destination o^* to which all cells in D are directed. Copies of the cells in D directed to o^* must be transferred in different time slots to avoid conflicts. Thus, at least k time slots are necessary to transfer all the cells in D. As a consequence, the schedule $\mathcal{TSA}[R]$ is not complete. □

As a trivial consequence:

Lemma 2: Let R be a k-complex request set. Consider a $\mathcal{TSA}[R]$ on F; if there exists a set D s.t. $D \subseteq R, |D| = k$ and $|\mathcal{I}_{\mathcal{TSA}}(D)| < k$, then $\mathcal{TSA}[R]$ is not complete.

We now introduce a lemma that will be used in subsequent proofs.

Lemma 3: For a strictly positive integer-valued random variable X whose average value $E[X] = \mu$ does not exceed m, the $100/m$th percentile does not exceed m.

Proof: Since $\mu \leq m$, by definition

$$\begin{aligned} E[X] = \mu &= \sum_{i=1}^{\infty} i \Pr\{X = i\} \\ &= \sum_{i=1}^{m} i \Pr\{X = i\} + \sum_{m+1}^{\infty} i \Pr\{X = i\} \\ &\geq \sum_{i=1}^{m} \Pr\{X = i\} + (m+1) \sum_{m+1}^{\infty} \Pr\{X = i\} \\ &= \Pr\{X \leq m\} + (m+1)(1 - \Pr\{X \leq m\}) \end{aligned}$$

but since $\mu \leq m$

$$\Pr\{X \leq m\} + (m+1)(1 - \Pr\{X \leq m\}) \leq m$$

and finally, solving for $\Pr\{X \leq m\}$

$$\Pr\{X \leq m\} \geq \frac{1}{m}. \qquad \square$$

We also need the following result in subsequent proofs, which is a extension of the well-known combinatorics "pigeonhole" principle.

Lemma 4: If u transfer units (objects) are transferred in t time slots (containers), it is possible to find a subset C of containers, such that $|C| = c$ in which at least $\lceil ((uc)/t) \rceil$ objects are contained for each integer positive number $c \leq t$.

The main analytical result on frame-based IQ switches is in Theorem 3 of Section V, whose statement is rephrased below.

Theorem 8: Let R be a k-complex request set, such that $|R| = kN_a$. For any finite integer $S, S \geq 2$, there exists an integer N_0 such that $\forall N_a > N_0$, no complete $\mathcal{TSA}[R]$ exists with frame length $L \leq Sk$.

Note that S is equivalent to an internal switch speedup, since the k-complex request set has k cells per output and k cells per active input, hence, it can be transferred on the output lines in k time slots by an OQ switch. Thus, the theorem states that, if the number of active input ports is sufficiently large, no speedup value is sufficient to obtain a complete $\mathcal{TSA}$, hence, to achieve 100% throughput. This result will be separately proved in the cases $S = 2, N_0 = 32$, and $S = 3, N_0 = 2187$; the proofs can be extended to larger values of S.

Proof: [Case $S = 2$]. Since we consider in this proof the case $S = 2$, we must show that no complete $\mathcal{TSA}$ exists if $L \leq 2k$. Obviously, if no complete $\mathcal{TSA}$ exists for a frame length $L = 2k$, no complete $\mathcal{TSA}$ can be found for shorter frame lengths. Thus, we consider only the case $L = 2k$.

We will prove that, for any possible $\mathcal{TSA}[R]$, a subset G of R can be found, such that $|G| = k$ and $|\mathcal{I}_{\mathcal{TSA}}(G)| < k$. Then, for Lemma 2, we can state that the considered $\mathcal{TSA}[R]$ is not complete.

The identification of set G will require a number of steps, defining a chain of subsets of R: $H \subseteq G \subseteq C \subseteq B \subseteq R$, as well as a subset W of the time slots in the frame F.

- Set B comprises all cells whose transfer is scheduled either once or twice in the frame (not a larger number of times).
- Set W comprises a number of time slots smaller than $k/2$, chosen so as to maximize the number of cells in B transferred (possibly partially) within W.

- Set C comprises all cells whose transfer is scheduled at least once within W (but not more than twice, being $C \subseteq B$).
- Set H comprises all cells whose transfer is scheduled only within W (either once or twice, being $H \subseteq B$).
- Set G is obtained by adding cells to H (if necessary, see below) to obtain $|G| = k$.

It is possible to visualize these sets by considering a scheduling as a $L \times N$ matrix $\boldsymbol{S}$, where rows correspond to time slots in the frame F, and columns correspond to switch input ports in I. The elements of $\boldsymbol{S}$ are $s_{nl} = 0$ if no cell from i_l is scheduled for transfer in time slot f_n, and $s_{nl} = m$ if in time slot f_n the transfer of cell r_m is scheduled from i_l to a subset of its destinations D_{mk}. By so doing, the definition of the above sets becomes:

- B comprises all cells whose identifier appears either once or twice in $\boldsymbol{S}$ (but no more than twice)
- W comprises the rows of $\boldsymbol{S}$ with the largest numbers of elements of B (selecting at most $k/2 - 1$ rows)
- C comprises all cells of B whose identifier appears in the rows of $\boldsymbol{S}$ within W (the same identifier may appear also once outside W)
- H comprises all cells of B whose identifier appears only in the rows of $\boldsymbol{S}$ within W (either once or twice)

To obtain a complete $\mathcal{TSA}[R]$ it is necessary that $\mathcal{I}_{\mathcal{TSA}}(\{r_l\}) \neq \emptyset, \forall r_l \in R$; thus, we will consider only the class of $\mathcal{TSA}[R]$ for which $\mathcal{I}_{\mathcal{TSA}}(\{r_l\}) \neq \emptyset$, $\forall r_l \in R$.

If $L = 2k$, then the average size of the image of individual cells in R through $\mathcal{TSA}[R]$ cannot be greater than 2 (because the copies of kN_a cells must fit into $2kN_a$ time slots; considering matrix $\boldsymbol{S}$, at most all its $2kN_a$ elements can be nonzero, thus, scheduling on the average twice the transfer of each one of the kN_a cells), i.e.: $E_{r_l}[|\mathcal{I}_{\mathcal{TSA}}(\{r_l\})|] \leq 2$. As a consequence, from Lemma 3, $|\mathcal{I}_{\mathcal{TSA}}(\{r_l\})| \leq 2$ for at least half of the cells in R.

Let $B \subset R$ be the set of cells for which $|\mathcal{I}_{\mathcal{TSA}}(\{r_l\})| \leq 2, \forall r_l \in B$; for Lemma 3, $|B| \geq |R|/2 = kN_a/2$. Thus, the average number of cells in B that are scheduled in each time slot of the frame is never less than $(|B|/L) \geq ((kN_a)/2)(1/(2k)) = ((N_a)/4)$.

For Lemma 4, there exists a set W of time slots with $|W| < k/2$ in which a set C of cells belonging to B (i.e., $C \subseteq B$) is scheduled, and $|C| \geq |W|N_a/4$.

Let us now restrict the attention to the case $|C| \geq 4k$, which implies that $N_a > 32$, and that k must be sufficiently large to generate enough cells to significantly fill the frame with transfers.

Let H be the set of cells $r_l \in C$ for which $\mathcal{I}_{\mathcal{TSA}}(\{r_l\}) \subseteq W$. Note that H comprises all the cells in C for which $|\mathcal{I}_{\mathcal{TSA}}(\{r_l\})| = 1$, and that H may be a null set.

Now we consider three cases:

1) $|H| \geq k$. We can set $G = H$, obtaining $|G| \geq k$ and $|\mathcal{I}_{\mathcal{TSA}}(G)| < k$.
2) $k/2 \leq |H| < k$. Let G be a set of cells such that $H \subseteq G \subset C, |G| = k$. We define a set of time slots $X = \mathcal{I}_{\mathcal{TSA}}(G \setminus H) \cap W$. Since $\mathcal{I}_{\mathcal{TSA}}(H) \subseteq W$, by construction

$$\mathcal{I}_{\mathcal{TSA}}(G) \subseteq W \cup [\mathcal{I}_{\mathcal{TSA}}(G \setminus H) \setminus X]$$
$$|\mathcal{I}_{\mathcal{TSA}}(G)| \leq |W| + |\mathcal{I}_{\mathcal{TSA}}(G \setminus H) \setminus X|.$$

Since $\mathcal{I}_{\mathcal{TSA}}(r_l) \forall r_l \in (G \setminus H)$ has at most one element in $[\mathcal{I}_{\mathcal{TSA}}(G \setminus H) \setminus X]$

$$|\mathcal{I}_{\mathcal{TSA}}(G \setminus H) \setminus X| = \left| \bigcup_l \mathcal{I}_{\mathcal{TSA}}(\{r_l \,|\, r_l \in (G \setminus H)\}) \setminus X \right| \leq \sum |\mathcal{I}_{\mathcal{TSA}}(\{r_l \,|\, r_l \in (G \setminus H)\}) \setminus X| \leq k/2.$$

Since $|W| < k/2$, we have $|\mathcal{I}_{\mathcal{TSA}}(G)| < k/2 + k/2 = k$.
3) $0 \leq |H| < k/2$. Define $h = k - |H|$; obviously, $k/2 < h \leq k$. Each cell in $C \setminus H$ (at least $3k + h$) will be scheduled once in $F \setminus W$ (referring to matrix $\boldsymbol{S}$, the cells in $C \setminus H$ appear once in the rows within W and once in rows not belonging to W). On the average, at least $((3k + h)/(|F \setminus W|))$ cells belonging to $C \setminus H$ will be scheduled in each time slot in $F \setminus W$ (since we have set $|C| \geq 4k$, we have $|H| = k - h$, thus, $|C \setminus H| \geq 4k - (k - h)$). Note that $((3k + h)/(|F \setminus W|)) > (h/(k/2))$, because $3k + h \geq 4h$ and $|F \setminus W| < |F| = 2k$. Thus, for Lemma 4, it is possible to find a set of cells H', s.t. $H' \subseteq (C \setminus H), |H'| = h$, and $|\mathcal{I}_{\mathcal{TSA}}(H') \setminus (\mathcal{I}_{\mathcal{TSA}}(H') \cap W)| \leq (k/2)$ (referring to matrix $\boldsymbol{S}$, H' comprises the cells that belong to the $k/2$ rows not in W containing the largest numbers of cells in $C \setminus H$).

Finally, let $G = H \cup H'$. By construction, $|G| = k$ and

$$|\mathcal{I}_{\mathcal{TSA}}(G)| = |\mathcal{I}_{\mathcal{TSA}}(H)| + |\mathcal{I}_{\mathcal{TSA}}(H')| < k. \qquad \square$$

When the size of the switch grows, it is possible to generalize the proof for frame sizes which are integer multiples of k. We briefly sketch the proof for $L = 3k$.

Proof: [Case $S = 3$]. Also in this case we will prove that, for each $\mathcal{TSA}[R]$, a subset A of R can be found, such that $|A| = k$ and $|\mathcal{I}_{\mathcal{TSA}}(A)| < k$. Then, for Lemma 2 we conclude that the considered $\mathcal{TSA}[R]$ is not complete.

If $L = 3k$, the average size of the image of individual cells in R through $\mathcal{TSA}[R]$ cannot be larger than 3, i.e., $E_{r_l}[|\mathcal{I}_{\mathcal{TSA}}(\{r_l\})|] \leq 3$. As a consequence, from Lemma 3, $|\mathcal{I}_{\mathcal{TSA}}(\{r_l\})| \leq 3$ for at least one third of the cells in R.

Let $B \subset R$ be the set of cells for which $|\mathcal{I}_{\mathcal{TSA}}(\{r_l\})| \leq 3, \forall r_l \in B$; for Lemma 3, $|B| \geq |R|/3 = kN_a/3$. Thus, the average number of cells in B that are scheduled in each time slot of the frame is never less than $((|B|)/L) \geq ((kN_a)/3)(1/(3k)) = ((N_a)/9)$.

For Lemma 4, there exists a set W of time slots with $|W| < k/3$ in which a set C of cells belonging to B (i.e., $C \subseteq B$) is scheduled, and $|C| \geq |W|N_a/9$.

Let us now restrict the attention to the case $|C| \geq 81k$, which implies that $N_a > 2187$, and that k must be sufficiently large to generate enough cells to significantly fill the frame with transfers.

Let H be the set of cells $r_l \in C$ for which $\mathcal{I}_{\mathcal{TSA}}(\{r_l\}) \subseteq W$. Note that H comprises all the cells in C for which $|\mathcal{I}_{\mathcal{TSA}}(\{c_i\})| = 1$, and that H may be a null set.

For simplicity, we assume that $H = \emptyset$. The proof, however can be extended to the more general case. In this case each one of the $|C|$ cells in C will be scheduled once or twice in the residual time slots belonging to $F \setminus W$.

On the average, $((|C|)/(|F \setminus W|))$ cells belonging to C will be scheduled in each time slot belonging to $F \setminus W$. Under the assumption $|C| \geq 81k, ((|C|)/(|F \setminus W|)) > 27$. Thus, for Lemma 4, there exists a set of W' time slots, with $W' = k/3$, in which at least $9k$ cells in C are scheduled. Let Y the set comprising these cells.

Let H' be the set of cells $r_l \in Y$ for which $\mathcal{I}_{\mathcal{TSA}}(\{r_l\}) \subseteq W \cup W'$. Note that H' comprises all the cells in Y for which $|\mathcal{I}_{\mathcal{TSA}}(\{r_l\})| \leq 2$. Again, we suppose that $H' = \emptyset$.

In this case each one of the $|Y|$ cells in Y will be scheduled once in the residual time slots belonging to $F \setminus (W \cup W')$. On average $((|Y|)/(|F \setminus (W \cup W')|))$ cells belonging to Y will be scheduled in each time slot belonging to $F \setminus (W \cup W')$. Note that $((|Y|)/(|F \setminus (W \cup W')|)) > 3$. Thus, for Lemma 4, there exists a set of W'' time slots, with $W'' = k/3$, in which at least k cells in Y are scheduled. Let Y' be the set comprising these cells. Since $|Y'| = k$ and $\mathcal{I}_{\mathcal{TSA}}(Y') \subseteq (W \cup W' \cup W'')$, we get $|\mathcal{I}_{\mathcal{TSA}}(Y')| < k$. □

REFERENCES

[1] M. Ajmone Marsan, A. Bianco, E. Filippi, P. Giaccone, E. Leonardi, and F. Neri, "On the behavior of input queuing switch architectures," *Eur. Trans. Telecommun.*, vol. 10, no. 2, pp. 111–124, Mar./Apr. 1999.

[2] M. Guo and R. Chang, "Multicast ATM switches: Survey and performance evaluation," *Comput. Commun. Rev.*, vol. 28, no. 2, pp. 98–131, Apr. 1998.

[3] Z. Liu and R. Righter, "Scheduling multicast input-queued switches," *J. Scheduling*, pp. 99–114, May 1999.

[4] X. Chen, I. Lambadaris, and J. Hayes, "A general unified model for performance analysis of multicast switching," in *Proc. IEEE GLOBECOM*, vol. 3, 1992, pp. 1498–502.

[5] J. F. Hayes, R. Breault, and M. K. Mehmet-Ali, "Performance analysis of a multicast switch," *IEEE Trans. Commun.*, vol. 39, pp. 581–587, Apr. 1991.

[6] J. Hui and T. Renner, "Queueing strategies for multicast packet switching," in *Proc. IEEE GLOBECOM'90*, 1990, pp. 1431–1437.

[7] C. K. Kim and T. T. Lee, "Performance of call splitting algorithms for multicast traffic," in *Proc. IEEE INFOCOM*, 1990, pp. 348–356.

[8] M. Andrews, S. Khanna, and K. Kumaran, "Integrated scheduling of unicast and multicast traffic in an input-queued switch," in *Proc. IEEE INFOCOM*, vol. 3, 1999, pp. 1144–1151.

[9] S. T. Chuang, A. Goel, N. McKeown, and B. Prabhakar, "Matching output queueing with a combined input/output-queued switch," *IEEE J. Select. Areas Commun.*, vol. 17, pp. 1030–39, June 1999.

[10] G. Nong and M. Hamdi, "On the provision of integrated QoS guarantees of unicast and multicast traffic in input-queued switches," in *Proc. IEEE GLOBECOM*, vol. 3, 1999, pp. 1742–1746.

[11] M. Ajmone Marsan, A. Bianco, P. Giaccone, E. Leonardi, and F. Neri, "On the throughput of input-queued cell-based switches with multicast traffic," in *Proc. IEEE INFOCOM*, Anchorage, AK, Apr. 2001, pp. 1664–1672.

[12] ——, "Optimal multicast scheduling in input-queued switches," in *Proc. IEEE Int. Conf. Communications*, Helsinki, Finland, June 2001, pp. 2021–2027.

[13] N. McKeown and B. Prabhakar, "Scheduling multicast cells in an input-queued switch," in *Proc. IEEE INFOCOM*, vol. 1, San Francisco, CA, Mar. 1996, pp. 261–278.

[14] M. G. Hluchyj, M. J. Karol, and S. Morgan, "Input versus output queueing on a space division switch," *IEEE Trans. Commun.*, vol. COM-35, pp. 1347–1356, Dec. 1987.

[15] M. A. Marsan, E. Leonardi, M. Mellia, and F. Neri, "On the stability of input-queued switches with speedup," *IEEE/ACM Trans. Networking*, vol. 9, pp. 104–118, Feb. 2001.

[16] P. R. Kumar and S. P. Meyn, "Stability of queueing networks and scheduling policies," *IEEE Trans. Automat. Contr.*, vol. 40, pp. 251–260, Feb. 1995.

[17] N. McKeown, A. Mekkittikul, V. Anantharam, and J. Walrand, "Achieving 100% throughput in an input-queued switch," *IEEE Trans. Commun.*, vol. 47, pp. 1260–1272, Aug. 1999.

[18] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Trans. Automat. Contr.*, vol. 37, pp. 1936–1948, Dec. 1992.

Marco Ajmone Marsan (F'99) holds degrees in electronic engineering from the Politecnico di Torino, Torino, Italy, and the University of California at Los Angeles (UCLA). In 2002, he was awarded an *Honoris Causa* degree in telecommunication networks from the Budapest University of Technology and Economics, Budapest, Hungary.

He is currently a Full Professor in the Department of Electronics, Politecnico di Torino. From 1975 to 1987, he was with the Department of Electronics, Politecnico di Torino, first as a Researcher, then as an Associate Professor. From 1987 to 1990, he was a Full Professor in the Department of Computer Science, University of Milan, Milan, Italy. During the summers of 1980 and 1981, he was with the Research in Distributed Processing Group, Department of Computer Science, UCLA. During the summer of 1998, he was an Erskine Fellow in the Department of Computer Science, University of Canterbury, Christchurch, New Zealand. He has coauthored over 300 journal and conference papers in the areas of communications and computer science, as well as two books, *Performance Models of Multiprocessor Systems* (Cambridge, MA: MIT Press) and *Modeling With Generalized Stochastic Petri Nets* (New York: Wiley). His current research interests are in the fields of performance evaluation of communication networks and their protocols.

Dr. Marsan received the Best Paper Award of the Third International Conference on Distributed Computing Systems, Miami, FL, in 1982. He participates in a number of Editorial Boards of international journals, including the IEEE/ACM TRANSACTIONS ON NETWORKING.

Andrea Bianco (M'98) was born in Torino, Italy, in 1962. He received the Dr.Ing. degree in electronics engineering and the Ph.D. degree in telecommunications engineering from Politecnico di Torino, Torino, Italy, in 1986 and 1993, respectively.

He is currently an Associate Professor in the Department of Electronics, Politecnico di Torino. From 1994 to 2001, he was an Assistant Professor with the Politecnico di Torino, first in the Production Systems Department, later in the Department of Electronics. In 1993, he was with Hewlett-Packard Laboratories, Palo Alto, CA. In the summer of 1998, he was with the Department of Electronics, Stanford University, Stanford, CA. He has co-authored over 80 papers published in international journals and presented in leading international conferences in the area of telecommunication networks. His current research interests are in the fields of protocols for all-optical networks and switch architectures for high-speed networks.

Dr. Bianco has participated in the technical program committees of several conferences, including the IEEE Infocom 2000, IFIP ONDM (Optical Network Design and Modeling) 2002 and 2003, and Networking 2002. He is the Technical Program Co-Chair of the High Performance Switching and Routing 2003 Workshop.

Paolo Giaccone (S'99–M'02) received the Dr.Ing. and Ph.D. degrees in telecommunications engineering from the Politecnico di Torino, Torino, Italy, in 1998 and 2001, respectively.

He is currently an Assistant Professor in the Department of Electronics, Politecnico di Torino. During the summer of 1998, he visited the High Speed Networks Research Group, Bell Labs, Lucent Technologies, Holmdel, NJ. During 2000–2001 and summer 2002, he was with the Department of Electrical Engineering, Stanford University, Stanford, CA. Between 2001 and 2002 he held a Postdoctoral position in the Department of Electronics, Politecnico di Torino, and during summer 2002, at Stanford University, Stanford, CA. His main area of interest is the design of scheduling policies for high-performance routers.

Emilio Leonardi (M'99) received the Dr.Ing degree in electronics engineering and the Ph.D. degree in telecommunications engineering from the Politecnico di Torino, Torino, Italy, in 1991 and 1995, respectively.

He is currently an Assistant Professor in the Department of Electronics, Politecnico di Torino. In 1995, he was with the Department of Computer Science, University of California at Los Angeles. In the summer of 1999, he was with the High Speed Networks Research Group, Lucent Technology–Bell Labs, Holmdel, NJ, and in the summer of 2001, he was with the Department of Electrical Engineering, Stanford University, Stanford, CA. He has coauthored over 100 papers published in international journals and presented in leading international conferences. His areas of interest are all-optical networks, queueing theory, and scheduling policies for high-speed switches.

Fabio Neri (M'99) was born in Novara, Italy, in 1958. He received the Dr.Ing. and Ph.D. degrees in electrical engineering from the Politecnico di Torino, Torino, Italy, in 1981 and 1987, respectively.

He is currently a Full Professor in the Department of Electronics, Politecnico di Torino. His teaching duties include graduate-level courses on computer communication networks and on the performance evaluation of telecommunication systems. He leads a research group on optical networks at the Politecnico di Torino. He has recently been involved in several European projects on WDM networks, including the ACTS project SONATA, which envisaged a single-layer optical transport network encompassing all concentration, distribution, transmission, switching and routing functions within a national network, and the IST project DAVID, which investigates the potential of optical packet switching in metropolitan and backbone networks. He coordinated the participation of his research group to several national Italian research projects. He has coauthored over 100 papers published in international journals and presented in leading international conferences. His research interests are in the fields of performance evaluation of communication networks, high-speed and all-optical networks, packet-switching architectures, discrete event simulation, and queueing theory.

Dr. Neri is a Member of the IEEE Communications Society. He has served on the boards of several IEEE conferences and journals, and participated in the Technical Program Committees of several conferences, including the IEEE Infocom and the IEEE Globecom. He was General Co-Chair of the 2001 IEEE Local and Metropolitan Area Networks Workshop and of the 2002 IFIP Working Conference on Optical Network Design and Modeling.