

Packet Scheduling in Input-Queued Cell-Based Switches

M. Ajmone Marsan, A. Bianco, P. Giaccone, E. Leonardi, F. Neri

Abstract—Input-queued switch architectures play a major role in the design of high performance switches and routers for packet networks. These architectures must be controlled by a scheduling algorithm, which solves contentions in the transfer of data units from inputs to outputs. Several scheduling algorithms were proposed in the literature for input-queued cell switches, operating on fixed-size data units. In this paper we consider the case of packet switches, i.e., devices operating on variable-size data units at their interfaces, but internally operating on cells, and we propose novel extensions of known scheduling algorithms. We prove that the maximum throughput achievable by input-queued packet switches is identical to that achievable with input- and output-queued cell switches. We show by simulation that, in the case of packet switches, input-queued architectures may provide performance advantages over output-queued architectures.

Keywords—IQ switch, packet switching.

I. INTRODUCTION

Most of the old designs of switching fabrics, i.e., of the part of a switch (or router) that is in charge of the transfer of data units from input to output line interfaces, assumed an output-queued (OQ) architecture. However, input-queued (IQ) switch architectures [1] have recently received increasing attention, and they are currently considered by many designers as the best solution when the line speed is pushed to technological limits.

With respect to IQ schemes, OQ has the advantage that delays through the switch can be more easily controlled, and the implementation of (concentrated) fair queuing algorithms at the output is well understood [2], [3], [4], [5]. However, a weak point of OQ is that both the switching fabric and the output queues in line cards must operate at a speed equal to the sum of the rates of all input lines. In applications where the number of interfaces is large and/or the line rate is high, this makes OQ impractical. Instead, IQ schemes permit all the components of the switch (input interfaces, switching fabric, output interfaces) to operate at a speed which is compatible with the data rate of input/output lines.

One of the reasons why IQ was almost ruled out in the past by switching fabrics designers is the performance reduction due to head-of-the-line blocking in the case of a single queue per input interface [6]. Virtual Output Queuing schemes [1], [7], [8], [9] are today adopted to practically overcome this problem: in each interface card, input buffers are organized into a set of queues, each queue storing data directed toward a specific output interface.

A major issue in the design of IQ switches is that the access to the switching fabric must be controlled by some form of scheduling algorithm to avoid contention. Note that the term “scheduling algorithm” for switching architectures is used in the literature for two different types of schedulers: switching ma-

trix schedulers and flow-level schedulers [10]. *Switching matrix schedulers* decide which input interface is enabled to transmit in an IQ switch; they avoid blocking and solve contentions within the switching fabric. *Flow-level schedulers* decide which data flows must be served in accordance to QoS requirements. In this paper the term scheduling algorithm is only used to refer to the first class of algorithms. Several scheduling algorithms for IQ switches operating on fixed-size data units were proposed and compared in the literature (see, for example, [9], [11], [12], [13], [14], [15]).

The interest for the case of fixed-size data units is due to the fact that, in several advanced Internet routers, the switching fabric internally operates on cells, and input IP datagrams are internally segmented into ATM-like cells that are transferred to output interfaces, where they are reassembled into variable-size IP datagrams. Examples of cell-based routers and switches adopting switching fabrics with input buffers can be found both in commercial products and laboratory prototypes: the Lucent GRF [16], the Cisco GSR [17], the Tiny-Tera [18], the AN2/DEC [1], [19], the iPoint [12], the MGR/BBN [20].

In this paper we consider some known scheduling algorithms and develop novel variations to deal with variable-size packets; more precisely, we constrain the scheduling algorithm to deliver contiguously all the cells deriving from the segmentation of the same packet. In other words, variable-size packets are transformed into “trains of cells”, and the transfer of the cells belonging to the same train is scheduled in such a way that they remain contiguous in the delivery to the output card, i.e., they are not interleaved with the cells of another train. This constraint permits savings in memory and complexity, since the reassembly of packets at the output becomes much easier.

The performance of the considered scheduling algorithms, both in the case of cell scheduling and in the case of packet scheduling, is studied by analysis and simulation. Although there is a common belief that dealing with variable-size data units brings about performance penalties, we prove in this paper that the maximum throughput achievable by IQ packet switches is identical to that achievable with IQ and OQ cell switches. For what regards packet delays, we show that the relative merits of switches operating in cell-mode and in packet-mode depend on traffic characteristics.

A common objection to packet-mode operation is that long packets can keep a pair of switch ports busy for long times, thereby causing undesirable delays to time-constrained packets. However, two considerations are in order on this issue: the problem is typical of all packet-switching networks (unless packet preemption is permitted), and it becomes negligible for the very large data rates to which the switching architectures considered in this paper are targeted (since transmission times become very small also for large packets).

The authors are with the Dipartimento di Elettronica, Politecnico di Torino, Italy. E-mail: {ajmone,bianco,giaccone,leonardi,neri}@polito.it. This work was supported in part by a research contract between CSELT and Politecnico di Torino, and in part by the Italian MURST MQOS Project.

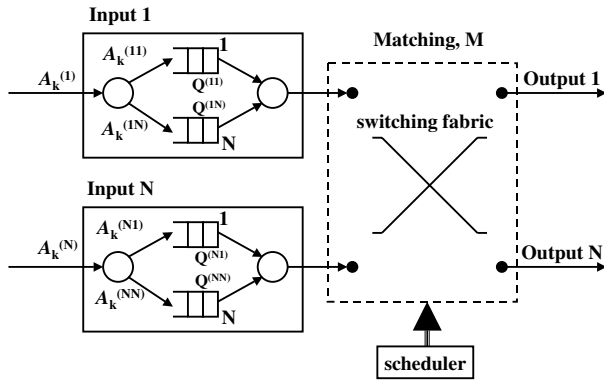


Fig. 1. Logical structure of an input-queued cell switch

The paper is organized as follows. Section II describes the logical switch architecture, both in the case of fixed-size and of variable-size packets. Section III overviews cell scheduling algorithms, and presents our modifications to turn them into packet scheduling algorithms. Section IV contains the proof that the maximum throughput achievable by IQ packet switches is identical to that achievable with IQ and OQ cell switches, and a simplified analytical model to study packet delays. Section V presents simulation results for cell and packet scheduling. Finally, Section VI concludes the paper.

II. LOGICAL ARCHITECTURE

In this section we describe the logical structure of cell and packet switches. Cell switches, i.e., switches operating on fixed-size data units, are introduced first; packet switches will then be built using cell switches as an internal building block.

A. Cell switches

A.1 Input queued cell switches

Fig. 1 shows the logical structure for an IQ cell switch. The switch operates on fixed-size data units, which can be ATM cells, or have any other convenient format. Borrowing from the ATM jargon, we use the term *cells* to identify the internal fixed-size data units.

Even if we shall refer to switches operating on fixed-size data units, our results are more generally applicable to any switch or router taking switching decision at equally-spaced time instants. The distance between two consecutive switching decisions is called *slot*, and the slot is the granularity in the allocation of switch resources.

We do not deal with the problem of partial slot filling due to the variable size of IP packets arriving at inputs, even if its impact on performance may be significant, depending on the slot size and the input packet length distribution.

The switch can have in general m inputs and n outputs, but we consider the more realistic case $m = n = N$ (indeed, one input and one output interface usually reside on the same “line card”). We also assume for simplicity that all input and output lines run at the same speed.

Packets are stored at input interfaces. Each input manages one FIFO queue for each output, hence a total of $N \times N = N^2$

queues are present. This queue separation permits to avoid performance degradations due to head-of-the-line blocking [21], and is called Virtual Output Queuing (VOQ) or Destination Queuing (DQ) [1], [9]. We assume that all packets belong to the same traffic class and limit our attention to unicast traffic.

Cells arrive at input i , $1 \leq i \leq N$, according to a discrete-time random process $A_k^{(i)}$, where k is a time slot index. At most one cell per slot arrives at each input, i.e., the data rate on input lines is no more than 1 cell per slot. When a cell with destination j arrives at input i , it is stored in the FIFO queue $Q^{(ij)}$. The number of cells in $Q^{(ij)}$ at time k is denoted by $X_k^{(ij)}$. These FIFO queues have limited capacity: each queue can store at most Q_{\max} cells.

We call $A_k^{(ij)}$ the arrival process at input i for output j ; the average cell arrival rate is denoted by $\lambda^{(ij)}$. The aggregation of all arrival processes at input i is $A_k^{(i)} = \{A_k^{(ij)}, 1 \leq j \leq N\}$. The aggregation of all arrival processes is $A_k = \{A_k^{(i)}, 1 \leq i \leq N\}$. A_k is termed *admissible* if no input and no output is overloaded, i.e., if $\sum_{i=1}^N \lambda^{(ij)} < 1, \forall j$ and $\sum_{j=1}^N \lambda^{(ij)} < 1, \forall i$. Otherwise A_k is *inadmissible*. For later use, we define the cell arrival rate matrix $\Lambda = [\lambda^{(ij)}]$, and the normalized cell arrival rate matrix $\Gamma = [\gamma^{(ij)}]$, with $\gamma^{(ij)} = \lambda^{(ij)} / (\sum_{n=1}^N \sum_{m=1}^N \lambda^{(nm)})$. We define also the average *packet* arrival rate, denoted by $\lambda_P^{(ij)}$. Similarly to Γ , we define the normalized packet arrival rate matrix $\Gamma_P = [\gamma_P^{(ij)}]$, which is the normalized packet arrival rate matrix.

The switching fabric is non-blocking and memoryless; at most one cell can be removed from each input and at most one cell can be transferred to each output in every slot. Since the speed at which cells are fed into output interfaces is equal to the speed at which cells are fed to input interfaces, we have a speed-up factor equal to 1. The scheduling algorithm decides which cells are transferred from the inputs to the outputs of the switch in every slot.

A.2 Output queued cell switches

The OQ cell switch needs no input buffers (neglecting buffers used at the input to store the newly arrived cell) because the switching fabric has enough capacity to transfer to the desired output all the cells received in one time slot. In the worst case (i.e., when a cell arrives at each input, and all cells are directed to the same output), this means that the bandwidth towards each output must be equal to the sum of the bandwidths available on all input lines. We say that the fabric must have a speed-up factor equal to N .

At each output, cells are stored in a single FIFO queue. For a fair comparison with IQ switches, we assume that the total amount of buffer space is kept constant, i.e., that the FIFO output queue can store $N \times Q_{\max}$ cells. This assumption gives some advantage to the OQ schemes, which can exploit some degree of buffer sharing (in IQ, we assume that Q_{\max} is the capacity of each virtual queue).

B. Packet switches (or routers)

Given the cell-switch logical architectures described in previous sections, we can now build a packet switch around them. We use as a reference the case in which a high-performance IP

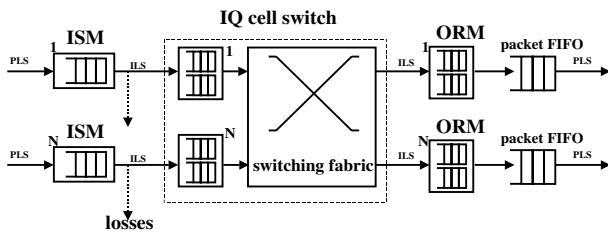


Fig. 2. Logical architecture for an IQ packet switch

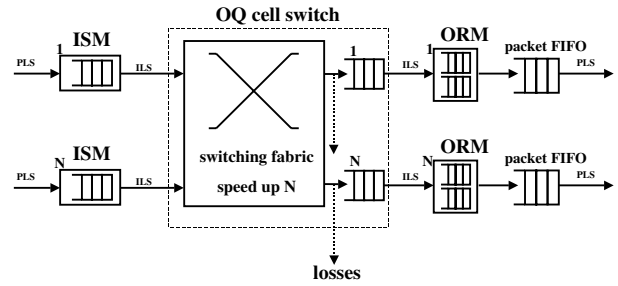


Fig. 3. Logical architecture for an OQ packet switch

router is built around an ATM cell-switch. Each router port has line interfaces where any data-link and physical layer protocol can be used to receive and transmit IP datagrams. The IP protocol sits on top of the data-link protocol at the input and at the output of the router. Within the IP layer at the input, routing functions are activated to associate an output port to the destination IP address. We neglect here issues related to the implementation of these functions, such as table look-up.

Input IP datagrams are segmented into ATM cells, that will be transferred to output ports by a high-performing ATM switching fabric. Once cells are delivered to an output port, they are reassembled into the IP datagram, which is transmitted on the output line according to possibly different line formats.

B.1 IQ packet switches

The logical architecture for an IQ packet switch is shown in Fig. 2. At each input an Input Segmentation Module (ISM) segments the incoming packet into cells. PLS is the external packet line speed. Since the ISM operates in store-and-forward mode, it must be equipped with enough memory to store a maximum-size packet, and the segmentation process starts only after the complete reception of the packet.

The cells resulting from the segmentation are transferred to the cell-switch input at a speed (called ILS) equal to the line speed PLS incremented to account for segmentation overheads. The capacity of input queues at the cell-switch is limited to Q_{\max} , hence losses can occur. We assume that the entire packet is discarded if the input queue of the cell-switch does not have enough free space to store all the cells deriving from the segmentation of the packet *when the first of these cells hits the queue*. This is of course a pessimistic assumption, but has the advantage of ease of implementation, and of avoiding the transmission of incomplete packet fractions through the switch.

The cell-based switching fabric transfers cells from input to output queues, according to a scheduling algorithm, as previously discussed. These cells are delivered to the Output Reassembly Module (ORM) at speed ILS. Here packets (i.e., IP datagrams) are reassembled. In general, cells belonging to different packets can be interleaved at the same output, hence more than one reassembly machine can be active in the same ORM. However, at most one cell reaches each ORM in a slot time, hence at most one packet is completed at each ORM in a slot.

Once a packet is complete, it is logically added to an output packet queue, called *packet FIFO* in the figure, from which packets are sequentially transmitted onto the output line. Note that the *packet FIFO* functionality is typically implemented by

imposing a sequential transfer from the suitable ORM to the output line of all the cells belonging to the same reassembled packet. No internal speedup is required to support ISM and ORM, but for compensating internal overheads.

It is possible to further simplify the structure of the switch, and to improve its performance, by enforcing additional constraints on the scheduling algorithm. Indeed, the cells belonging to the same packet are contiguous in the input queue of the internal cell-switch. As we shall see in Section III, it is possible to modify some well-known scheduling algorithms in such a way that, once the transfer through the switching fabric of the first cell of a packet has started towards the corresponding output port, no cells belonging to other packets can be transferred to that output. In such a way, cells belonging to the same packet are kept contiguous also in the output queue, and the ORM modules are not necessary any longer (or at most one per output is used). We call this class of scheduling algorithms *packet-mode scheduling*. Note that enforcing packet contiguity is not possible in OQ switches, where cell interleaving in output queues cannot be avoided, as we shall see.

If packet-mode scheduling is adopted in IQ packet switches, the logical architecture can be simplified by removing both the ORM module and the output packet FIFO from Fig. 2. Since each module operates in store-and-forward mode, hence introduces a delay equal to the packet size, the delays through the switch are reduced at least by a packet duration. Note also that the segmentation of the packet into cells is no longer required: the only necessary information is the (integer) number of slot times used to transfer the packet.

B.2 OQ packet switches

Fig. 3 shows the logical architecture of an OQ packet switch. Packets arrive at input ports where they are segmented by ISM modules, similarly to what happens in IQ routers. The cells obtained with the segmentation are sent to the cell-switch at speed ILS, and are immediately transferred to the output queues of the cell-switch, thanks to a speed-up equal to N in the switching fabric. Losses may occur at output queues, whose capacity is limited to $N \times Q_{\max}$ for each queue, since we are assuming the same buffering capacity for OQ and IQ switches.

From the output queues of the cell-switch, cells are delivered at speed ILS to an ORM module for reassembly. Once a packet is completed, it is queued in a packet FIFO queue similar to what was seen for the IQ case.

An important difference between OQ packet switches and IQ packet switches resides in the way of handling losses. Cells belonging to different packets can be interleaved in output queues. When a cell at the output of the switching fabric does not find room in the output queue of the cell-switch, it must be discarded. The other cells belonging to the same packet of the discarded cell may be in the output queue, or already in the ORM. We assume to be unable to identify and discard the other cells in the output queue: those cells will be discarded by the ORM module, which has knowledge of the existence of the packet. This means that cells belonging to packets that suffered partial losses unnecessarily use system resources.

The IQ architecture has the advantage that all the cells belonging to the same packet are contiguous in input queues, where losses occur. In the OQ case, losses occur in queues where cells of different packets are interleaved.

III. CELL AND PACKET SCHEDULING ALGORITHMS

A. Problem definition

This section describes scheduling algorithms. Remember that we used this term to refer to switching matrix schedulers, i.e., to the set of rules used to decide which input port is granted access to the switching fabric. Scheduling algorithms avoid blocking and solve contention within the switching fabric.

The scheduling algorithm selects a *matching* M , i.e., a set of input-output pairs with no conflicts, such that each input is connected with at most one output, and each output is connected with at most one input. In each slot, if input i is connected with output j , a cell is removed from $Q^{(ij)}$ and transferred to output j by properly configuring the non-blocking switching fabric.

The different IQ cell switch architectures that we shall discuss differ in their scheduling algorithms, hence in their matching algorithms. The need for good heuristic matching algorithms derives from the fact that the optimal solutions of the problem have very high complexity. The complexity is $O(N^3)$ for the Maximum Weight Matching (MWM) algorithm [22, Chapt. 8], that can be proved to yield the maximum achievable throughput using as metrics (weights) of input-output pairs either the number of cells to be transferred, or the time waited by the oldest cell; it is $O(N^{5/2})$ for the simpler and less efficient maximum size matching algorithm [23], which maximizes the number of input-output pairs in M . Well-known maximum size matching algorithms were proposed by Dinic [22] and Hopcroft [24].

B. The considered cell scheduling algorithms

A number of scheduling algorithms for IQ switch architectures have appeared in the technical literature. In this paper we consider four such proposals, namely iSLIP [9], [25], iOCF [9] (both with $\log N$ iterations), MUCS [12] (in its weighted version), and RPA [11] (in its static version). iSLIP has been chosen for its simplicity, iOCF due to its metrics based on cell age, RPA for both simplicity and good performance under unbalanced traffic patterns, and MUCS for the very good performance obtained in different traffic scenarios.

For the sake of brevity, we do not provide here a description of these algorithms. The reader is referred to the original works for detailed descriptions. We proposed general taxonomy for

scheduling algorithms in [15], where we classify accordingly the considered proposals.

C. Packet-mode scheduling algorithms

Packet-mode scheduling algorithms introduce the additional constraint of keeping the cells belonging to the same packet contiguous also in output queues. To achieve this, the scheduling algorithm must enforce that, once the transfer through the switching fabric of the first cell of a packet has started towards the corresponding output port, no cells belonging to other packets can be transferred to that output, i.e., when an input is enabled to transmit the first cell of a packet comprising k cells, the input/output connection must be enabled also for the following $k - 1$ slots. This is equivalent to have an infinite weight on the corresponding connection. Note that no conflicts can arise between infinitely-weighted connections, since no more than one cell can reach a given output in one slot, hence two connections directed to the same output cannot simultaneously have an infinite weight.

We propose to extend the four considered scheduling algorithms to operate in packet-mode. The only complexity increase in the implementation is to store a boolean variable at each input to flag over-prioritized connections.

IV. ANALYTICAL MODELS OF IQ PACKET SWITCHES

In this section we present analytical models of IQ packet switches. We first formally prove that the maximum throughput achievable by IQ packet switches operating in packet mode is identical to that achievable with IQ and OQ cell switches. This means that packet mode operation can achieve 100 % throughput provided the input traffic is admissible. We then show, with a simplified model, that although relative delays of cell- and packet- switches depend on traffic characteristics, it is possible to define traffic conditions in which packet switches provide lower delays than cell switches.

A. Maximum Achievable Throughput

A.1 Definitions and preliminary results

Given a system of M (in our case $M = N^2$) discrete-time queues of infinite capacities, let X_n be the row vector of queue lengths at time n ; i.e., $X_n = (x_n^1, x_n^2, \dots, x_n^M)$, where x_n^i is the number of customers in queue i at time n .

The evolution of the length of queue i is described by the expression $x_{n+1}^i = x_n^i + a_n^i - d_n^i$, where a_n^i represents the number of customers arrived at queue i in time interval $(n, n + 1]$, and d_n^i represents the number of customers departed from queue i in time interval $(n, n + 1]$. Let $A_n = (a_n^1, a_n^2, \dots, a_n^M)$ be the vector of the numbers of arrivals at all queues, and $D_n = (d_n^1, d_n^2, \dots, d_n^M)$ be the vector of the numbers of departures from all queues. With this notation, the system evolution equation can be written as $X_{n+1} = X_n + A_n - D_n$. We assume in this section that the entries a_n^i of vectors A_n are independent and identically distributed for variable n with fixed i , and independent for variable i with fixed n , although this latter constraint could be partly relaxed.

We indicate with $\|Y\|$ the Euclidean norm of vector $Y = (y^1, y^2, \dots, y^K)$: $\|Y\| = \sqrt{\sum_{i=1}^K (y^i)^2}$.

Definition 1: A system of queues is said to be **strongly stable** if $\lim_{n \rightarrow \infty} \sup E \|X_n\|$ is finite.

We assume that the stochastic process describing the evolution of the system of queues is an irreducible Discrete Time Markov Chain (DTMC), whose state vector at time n is $Y_n = (X_n, K_n)$, $Y_n \in \mathbb{N}^{M''}$, $X_n \in \mathbb{N}^M$, $K_n \in \mathbb{N}^{M'}$ and $M'' = M + M'$. Y_n is the combination of the queue length vector X_n and a vector K_n of integer parameters. Most systems of discrete-time queues of practical interest can be described with models that fall in the DTMC class. The following general criterion for the strong stability of systems falling into this class is therefore useful.

Theorem 1: Given a system of queues with state vector $Y_n = (X_n, K_n)$, and a function $V(X_n) = X_n W X_n^T$ (called Lyapunov function), if there exists a symmetric copositive¹ matrix $W \in \mathbb{R}^{M \times M}$, and two positive real numbers $\epsilon \in \mathbb{R}^+$ and $B \in \mathbb{R}^+$, such that:

$$E[V(X_{n+1}) - V(X_n) | Y_n] < -\epsilon \|X_n\| \quad \forall Y_n : \|X_n\| > B$$

then the system of queues is strongly stable. In addition, all the polynomial moments of queue lengths distributions are finite.

This is a re-phrasing of the results presented in [26, Sect.IV], to which readers are referred for a proof.

Being the identity matrix I a symmetric positive semidefinite matrix, hence a copositive matrix, it is possible to conclude that:

Corollary 1: Given a system of queues with state vector $Y_n = (X_n, K_n)$, if there exists $\epsilon \in \mathbb{R}^+$, $B \in \mathbb{R}^+$ such that:

$$E[X_{n+1} X_{n+1}^T - X_n X_n^T | Y_n] < -\epsilon \|X_n\| \quad \forall Y_n : \|X_n\| > B$$

then the system of queues is strongly stable, and all the polynomial moments of queue lengths distributions are finite.

A.2 Stability of packet-based IQ switches

Consider an IQ packet switch, and suppose that all input packet lengths are multiples of some unit length called UL (UL may correspond to a bit, a byte, or a cell). Consider the system of discrete-time queues comprising all input queues of the packet switch. The discrete time unit corresponds to a continuous time increment of size UL.

We assume that customers correspond to cells to be transferred from input to output ports. Since we consider an IQ switch, $d_n^i \in \{0, 1\}$, $\forall i$ and $\forall n$. The arrival of a packet corresponds to the arrival of a group of customers, whose cardinality equals the packet length in UL units. a_n^i can therefore be larger than 1. However, if the traffic is admissible, $E[a_n^i] \leq 1$, $\forall i$.

Let $t_n \in \mathbb{N}^+$ be a non-defective sequence of regeneration instants (or stopping times) for the evolution of the system of queues, i.e., for any t_n , the evolution of the system following t_n is independent of the evolution of the system before t_n ; moreover, $z_n = t_{n+1} - t_n$ are random variables such that $E[z_n] < \infty$ and $E[z_n^2] < \infty$, as we shall see.

Definition 2: An IQ packet switch follows a *renewal MWM schedule* if at each stopping time t_n a new switching configuration is selected according to the outcome of a MWM algorithm whose weights are proportional to queue lengths at time t_n , and the switching configuration is kept constant until t_{n+1} .

¹An $M \times M$ matrix Q is copositive if $XQX^T \geq 0 \quad \forall X \in \mathbb{R}^{+M}$.

Lemma 1: An IQ packet switch following a renewal MWM schedule is stable under any admissible i.i.d. input traffic pattern A_n such that $E[A_n A_n^T] < \infty \quad \forall n$.

Proof: The evolution of the system of discrete-time queues in the IQ packet switch is represented by a DTMC whose state is defined by the vector of queue lengths X_{t_n} ; between consecutive stopping times, the system evolution satisfies the following equation:

$$X_{t_{n+1}} = X_{t_n} + \sum_{i=0}^{z_n-1} (A_{t_n+i} - D_{t_n+i})$$

Note that all D_{t_n+i} , $i < z_n$ refer to the same switching configuration; however, they need not be all equal, since some queue scheduled for transmission at time t_n may become empty before the next stopping time. If this happens, no packet can be transferred from empty queues.

By using the Lyapunov function $V(X_{t_n}) = X_{t_n} X_{t_n}^T$:

$$\begin{aligned} E[V(X_{t_{n+1}}) | X_{t_n}] - V(X_{t_n}) &= \\ &= E \left[2 \sum_{i=0}^{z_n-1} (A_{t_n+i} - D_{t_n+i}) X_{t_n}^T + \right. \\ &\quad \left. + \sum_{i=0}^{z_n-1} (A_{t_n+i} - D_{t_n+i}) \sum_{i=0}^{z_n-1} (A_{t_n+i} - D_{t_n+i})^T \right] \end{aligned}$$

Thus, under the assumption that $E[A_n A_n^T]$ is finite (which corresponds to assuming finite packet length variances), since also $E[D_{t_n+i} D_{t_n+i}^T]$ is finite:

$$\begin{aligned} \lim_{\|X_n\| \rightarrow \infty} \frac{E[V(X_{t_{n+1}}) | X_{t_n}] - V(X_{t_n})}{\|X_{t_n}\|} &= \\ &= \lim_{\|X_n\| \rightarrow \infty} \frac{2E[\sum_{i=0}^{z_n-1} (A_{t_n+i} - D_{t_n+i}) X_{t_n}^T]}{\|X_{t_n}\|} \end{aligned}$$

Define now D_δ as the difference between $\sum_{i=0}^{z_n-1} D_{t_n+i}$ and $z_n D_{t_n}$; as noted before, this difference is due to the fact that some queues may become empty before changes in the switch configuration. Thus:

$$\begin{aligned} \frac{E[\sum_{i=0}^{z_n-1} (A_{t_n+i} - D_{t_n+i}) X_{t_n}^T]}{\|X_{t_n}\|} &= \\ &= \frac{E[\sum_{i=0}^{z_n-1} A_{t_n+i} X_{t_n}^T - z_n D_{t_n} X_{t_n}^T + D_\delta X_{t_n}^T]}{\|X_{t_n}\|} \end{aligned}$$

Wald's equation [27, §2-13] can be applied, since t_n is a sequence of stopping times, thus obtaining:

$$\begin{aligned} \frac{E[\sum_{i=0}^{z_n-1} A_{t_n+i} X_{t_n}^T - z_n D_{t_n} X_{t_n}^T + D_\delta X_{t_n}^T]}{\|X_{t_n}\|} &= \\ &= \frac{E[z_n] (E[A_n] - D_{t_n}) X_{t_n}^T + E[D_\delta] X_{t_n}^T}{\|X_{t_n}\|} \end{aligned}$$

Note that $E[D_\delta] X_{t_n}^T \leq M E[z_n^2]$, since at most M components of D_δ can be non-null, no component of D_δ can exceed the value z_n , and, finally, a component of D_δ can be non-null only if the corresponding queue length at time t_n is smaller than z_n . Moreover, for each admissible load and non-null queue length vector,

$(E[A_n] - D_{t_n})X_{t_n}^T < 0$ as proved in [28]. Thus:

$$\begin{aligned} \lim_{\|X_n\| \rightarrow \infty} \frac{E[V(X_{t_{n+1}}) | X_{t_n}] - V(X_{t_n})}{\|X_{t_n}\|} &= \\ &= \lim_{\|X_n\| \rightarrow \infty} \frac{2E[z_n](E[A] - D_{t_n})X_{t_n}^T + 2E[D_\delta]X_{t_n}^T}{\|X_{t_n}\|} = \\ &= 2E[z_n] \lim_{\|X_n\| \rightarrow \infty} \frac{(E[A] - D_{t_n})X_{t_n}^T}{\|X_{t_n}\|} < -E[z_n]\epsilon \end{aligned}$$

Definition 3: An IQ packet switch follows a *packet MWM schedule* if a new switching configuration is selected according to a MWM algorithm, whose weights are proportional to queue lengths, whenever either:

- all packet transmissions end at the same time, or
- all the queues selected for transfer become empty.

Lemma 2: Consider an IQ packet switch, following a *packet MWM schedule*, whose input traffic is formed by variable length packets with i.i.d. random size. Packet sizes are expressed in integer multiples of UL. Assume that at queue k the average packet size is l_k and the packet size variance is σ_k^2 (both being finite). Assume that the transmission of packets from all queues selected by the MWM algorithm starts at the same time with exactly the same rate. Consider the sequence of instants t_n at which either the transmission of all the packets at the head of the selected queues ends at the same time, or all selected queues become empty. The sequence t_n is a non-defective renewal process, and $z_n = t_{n+1} - t_n$ are such that $E[z_n] < \infty$ and $E[z_n^2] < \infty$.

Proof: For simplicity, assume that the packet length distributions at all queues are aperiodic, i.e., the maximum common divisor of all possible packet lengths expressed in UL is equal to 1. The proof can be easily extended to the case of periodicity. Each sequence of instants at which transmissions of packets end at queue k forms a discrete-time aperiodic renewal point process, thanks to the independence of packet lengths. Thus, for Blackwell's theorem [27, §2-19], the average number $E[f_n^k]$ of packets whose transmissions end at queue k at time n satisfies the following equation:

$$\lim_{n \rightarrow \infty} E[f_n^k] = 1/l_k \quad (1)$$

However, no more than one packet transmission can end at each queue at each time (assuming no packet is of length zero); thus $E[f_n^k]$ equals the probability that a packet ends:

$$E[f_n^k] = P\{\text{transmission ends at time } n \text{ and at queue } k\}$$

Limit (1) implies that, for any integer $m > 1$, there exists an instant n_k such that, $\forall n > n_k$:

$$P\{\text{transmission ends at time } n \text{ and at queue } k\} > \frac{1}{ml_k} > 0$$

The probability that at instant n the transmission of packets at the head of all queues selected for transmission (be their number N_s) ends can be easily computed, since no correlation exists

among queues behavior. Thus, given m , for $n > n_k, \forall k$:

$$\begin{aligned} P\{\text{all tx end at time } n\} &= \\ &= \prod_{k=1}^{N_s} P\{\text{tx ends at time } n \text{ and at queue } k\} > \\ &> \prod_{k=1}^{N_s} \frac{1}{ml_k} > 0 \end{aligned}$$

Consider now the sequence of instants t_n at which either all packet transmissions end, or selected queues become empty. The sequence t_n forms a renewal process; thus Blackwell's theorem applies:

$$P\{\text{all tx end at time } n\} = E[f_n] = 1/E[z_n]$$

where $E[f_n]$ is the average number of regenerations at time n ; since $P\{\text{all tx end at time } n\} > 0$, $E[z_n] < \infty$.

To prove that also $E[z_n^2] < \infty$, consider all packets transmitted from queue k between two subsequent regenerations; let W be the number of such packets, and L_j be their lengths expressed in UL. We can write:

$$\begin{aligned} E[z_n^2] - E^2[z_n] &= E\left[\left(\sum_{j=1}^W (L_j - E[L_j])\right)^2\right] = \\ &= E\left[\sum_{j=1}^W (L_j^2 - E^2[L_j])\right] + E\left[\sum_{j=1}^W \sum_{\substack{i=1 \\ i \neq j}}^W (L_j L_i - E[L_j L_i])\right] \end{aligned}$$

The second term in the sum can be easily shown to be null by conditioning on the value of W ; it can thus be eliminated. As a consequence:

$$E[z_n^2] - E^2[z_n] = E\left[\sum_{j=1}^W L_j^2\right] - \sum_{j=1}^W E^2[L_j]$$

and by Wald's equation, since regeneration points are stopping times for the sequence L_j :

$$E[z_n^2] - E^2[z_n] = E[W]E[L_j^2] - E[W]E^2[L_j] = E[W]\sigma_k^2$$

Being $E[W]$ finite (otherwise $E[z_n]$ would be infinite), it results $E[z_n^2] < \infty$. ■

We can now state our main result.

Theorem 2: Any IQ packet switch following a *packet MWM schedule* is strongly stable, provided that

- the input traffic is admissible
- the input traffic is formed by variable length packets with i.i.d. random size having finite average and variance
- the transmission of packets from all queues selected by the MWM algorithm starts at the same time with the same rate.

The proof is trivial from Lemma 1 and Lemma 2.

Note that in Section III-C and in our simulation experiments we do not consider the MWM schedule of Theorem 2; we instead adopt a *constrained continuous heuristic matching*, i.e., in every slot, matching heuristics are used to approximate a MWM on the (reduced size) switch in which busy inputs and

outputs are removed. Theorem 2 shows in general that there exist scheduling algorithms that guarantee stability. Furthermore, the simulation results shown in Section V will not exhibit instabilities even for the constrained continuous heuristic matching approach.

B. Packet delay estimation

The relation between the average packet delay values in packet-mode and cell-mode schedulers can be understood with the help of a simplified analytical queuing model.

We focus on one output port, and on packets directed from the different inputs to that output port, and consider only the packet delay component due to virtual output queuing (thus disregarding the effect of segmentation and reassembly modules). To estimate the packet delay with a queuing model, we also have to disregard the output conflicts in the switch; thus, our estimates will be reasonably accurate only for low to medium traffic loads. The transfer toward the output port corresponds to the packet service, and packets are modeled as customers requiring a variable amount of service. As a further simplification, we describe the packet arrival process at the switch ingress with a Poisson process, without taking care of overlapping ingress times.

This simplified setting corresponds to an M/G/1 queue, where cell-mode scheduling can be paralleled to processor-sharing (PS) or round-robin (RR) service, since all packets directed to the considered output are simultaneously served (again because of low traffic), and packet-mode scheduling can be paralleled to FIFO service, since each packet is served separately, with no interleaving of cells of different packets.

We know from queueing theory [29] that the average delay $E[D_{PS}]$, in the case of PS service, is:

$$E[D_{PS}] = \frac{\rho E[S]}{1 - \rho}$$

where $E[S]$ is the average service time and ρ is the queue traffic (or utilization factor). Instead, for an M/G/1 queue with FIFO service, we know that:

$$E[D_{FIFO}] = \frac{\rho E[S]}{1 - \rho} \times \frac{1 + C_v^2}{2}$$

where C_v is the coefficient of variation of the service time.

Note that $E[D_{PS}]$ is equal to the average delay in a M/M/1 queue with FIFO service, so that $E[D_{PS}] = E[D_{FIFO}]$ for negative exponential distribution of packet lengths ($C_v = 1$).

We define the *packet-mode gain*, denoted G , as the ratio between the average packet delay experienced with cell-mode scheduling and the average packet delay experienced with packet-mode scheduling. As noted before, in our simplified analysis these average packet delays refer only to the waiting time in input queues, not comprising the delays due to segmentation and reassembly. From the simplified queuing model, G can be estimated as:

$$G = \frac{2}{(1 + C_v^2)}$$

Thus, the analytical model predicts gains for packet-mode scheduling in the case of packet length distributions with small

variance ($C_v < 1$), whereas cell-mode scheduling is expected to provide lower packet delays when the packet length variance is large ($C_v > 1$).

Note that similar results could have been obtained by modeling cell-mode operation with group arrivals, and by interpreting individual customers as cells, and groups as packets. Packet delays would in this case be equivalent to group delays.

V. SIMULATION RESULTS

To evaluate the performance of IQ switching architectures and to validate our analytical models, we conducted quite a large number of simulation experiments. We only report here selected results for packet switches with equal numbers ($N = 16$) of input/output interfaces, assuming that all input/output line rates are equal, and that only unicast traffic flows are present. In IQ switches, each input queue $Q^{(ij)}$ has finite length Q_{\max} ; when a cell directed to output j arrives at input i , and queue $Q^{(ij)}$ is full, the cell is lost. No buffer sharing among queues is allowed.

A. Traffic scenarios

Our simulation models do not explicitly describe the arrival of IP datagrams at the packet-switch inputs. We instead model the arrival of cell bursts at the inputs of the internal cell-switch. These cell bursts are assumed to originate from the segmentation of a packet.

The cell arrival processes at input i , $A_k^{(i)}$, are characterized with a two-state ON-OFF model. When the input port is in the ON state, a packet is being received. The number of slots spent in the ON state, i.e., the size in cells of the packet, is a discrete random variable $\Phi^{(ij)}$ for packets directed to output j . No cells are received in the OFF state. The number of slots spent in the OFF state is geometrically distributed with average $E_{\text{OFF}} = (1 - p)/p$. The parameter p is set so as to achieve the desired input load.

We consider the following traffic scenarios, which are described through their corresponding Γ and Γ_P matrices (see Section II-A.1), and $\Phi^{(ij)}$ random variables.

Uniform Scenario. In this case $\gamma^{(ij)} = \gamma_P^{(ij)} = 1/N^2$, $\forall i, j$ (uniform traffic). Packet lengths are uniformly distributed with lengths ranging between 1 and 192 cells. The maximum packet size comes from the Maximum Transmission Unit (MTU) of IP over ATM [30], which is equal to 9188 octets (9140 octets of user payload, and 20 + 20 octets of TCP and IPv4 headers, to which 8 octets are added for LLC/SNAP encapsulation). AAL5 then turns 9188 octets into 192 ATM cells.

Diagonal Scenario. In this case $\gamma_P^{(ij)} = 1/N^2$ (the traffic is uniform at the packet level), but all packets with equal index of input and output ports ($i = j$) have a fixed size equal to 100 cells, whereas the packet size always is equal to 3 cells for packets directed to an output port with different index from their input port ($i \neq j$). The reason for considering this scenario is to highlight the starvation effect in the service of short packets due to the transfer of long packets.

B. Performance indices

Results are presented with graphs where the following performance indices are plotted versus the switch traffic load. The

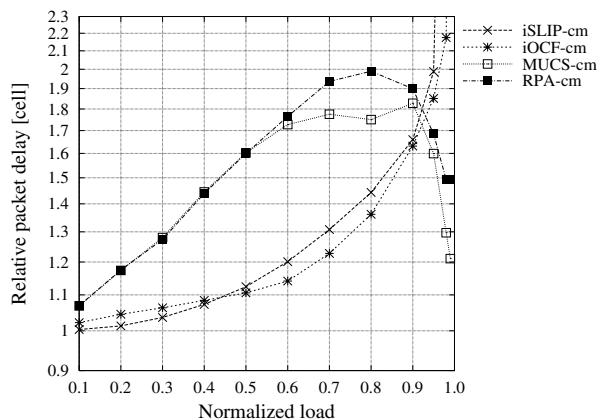


Fig. 4. Relative average packet delay for cell-mode scheduling in the uniform traffic scenario

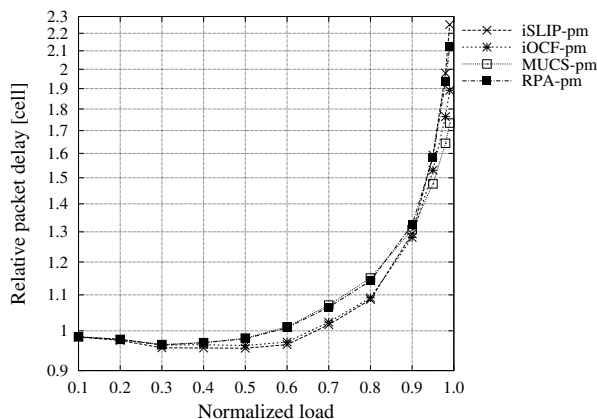


Fig. 5. Relative average packet delay for packet-mode scheduling in the uniform traffic scenario

latter is defined as the ratio between the input traffic load and the total capacity of input/output lines, both of them computed at the cell level (hence, the switch traffic load varies between 0 and 1).

- **Cell delay.** This is the time spent by cells in the cell-switch queues. For this metrics we shall consider the average value only.

- **Packet delay.** This is the overall delay of a packet, considering the ISM module, the internal cell-switch queues, the ORM module, and the final packet FIFO. It is computed only for packets completely delivered at switch outputs, measuring the time from the ingress of the last cell of the packet into the ISM module until the egress of that same last cell from the final packet FIFO. Constant delay components are removed: a single-cell packet traverses an empty packet switch in null time, and a packet comprising k cells has a best-case delay equal to $2(k - 1)$ slots, due to wait in the segmentation and reassembly phases. We shall consider only the average value of packet delay.

Since we want to compare IQ switch architectures, adopting either cell or packet scheduling, against OQ switches, we often consider *relative* (or *normalized*) performance indices, i.e., we divide the absolute value taken by a performance index in the case of IQ architectures by the value taken by the same performance index in an OQ switch loaded with the same traffic pattern.

Simulation runs were executed until the estimate of the average cell delay reached with probability 0.95 a relative width of the confidence interval equal to 2%. The estimation of the confidence interval width is obtained with the batch means approach.

C. Uniform Scenario

Fig. 4 shows, for the four considered scheduling algorithms operating in *cell-mode*, curves of the normalized average packet delay. Note that delay values are always larger than 1, i.e., IQ switches always yield longer delays, but differences are limited within a factor 2.5, for load smaller than 0.95. No losses were experienced with queue lengths equal to 30,000 cells.

The differences between the considered heuristics mainly depend on the different metrics.

Performance results are significantly different if the *packet-mode* scheduling proposed in this paper is considered. Curves of the relative average delays are shown in Fig. 5. Differences between the four algorithms are limited, and we observe (small) gains over OQ switches for loads up to around 0.6. IQ exhibits the largest delay advantage over OQ at loads around 0.4. To be fair in the comparison with OQ switches, we take into account, for IQ switches, delays due to ORM modules and to packet FIFOs, although these modules are not necessary in packet-mode operation, and at most one reassembly machine is active at each output, as discussed in Section II-B.1. Cell delays in this same scenario, not shown here, are larger for all IQ architectures than for OQ at all loads.

The reductions in packet delays are interesting, specially if we consider the negligible additional cost of implementing packet-mode schedulers. The simple queueing model of Section IV-B justifies this performance improvement, which is not very intuitive (in this case $C_v = 0.287$). Note that more complex scheduling algorithms, better tailored to the statistics of packet traffic, could probably provide even larger gains.

If the IQ switch architecture is further simplified to take advantage of packet-mode scheduling by removing ORM modules and output packet FIFOs, the gain in average packet delay over OQ becomes much larger, since the delay necessary to reassemble packets is avoided.

D. Diagonal Scenario

Figs. 6 and 7 show the absolute average packet delay curves for large packets when, respectively, cell-mode and packet-mode are adopted in the diagonal traffic scenario.

Only iSLIP experienced losses in cell-mode with queue lengths equal to 10,000 cells. When packet-mode is adopted, the total packet delay is always smaller than for cell-mode (except when losses were experienced).

Packet-mode scheduling obtains almost always delays smaller than OQ, whereas this is not true for cell-mode scheduling. This fact is due to the chosen packet length distribution, as foreseen by the model presented in Section IV-B. The delay improvement obtained with the use of packet-mode scheduling in the transfer of long packets is paid by shorter packets, as can

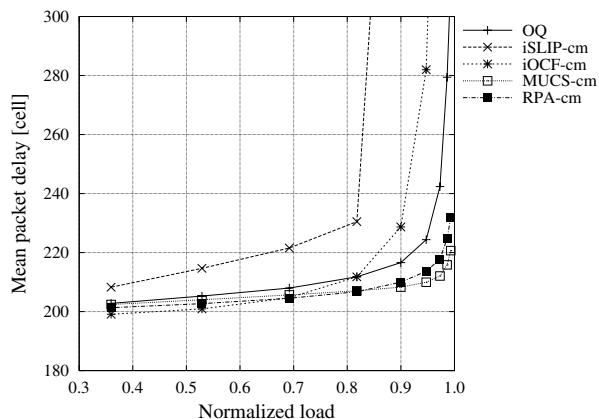


Fig. 6. Packet delay for large packets in the diagonal traffic scenario with cell-mode scheduling

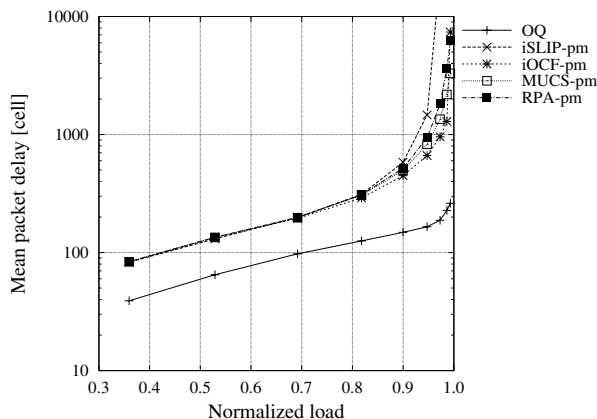


Fig. 8. Packet delay for short packets in the diagonal traffic scenario with packet-mode scheduling

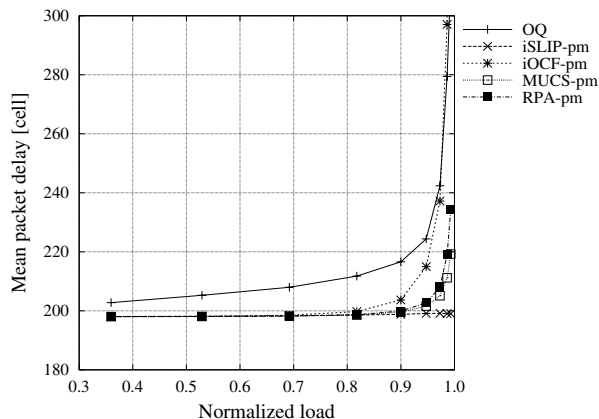


Fig. 7. Packet delay for large packets in the diagonal traffic scenario with packet-mode scheduling

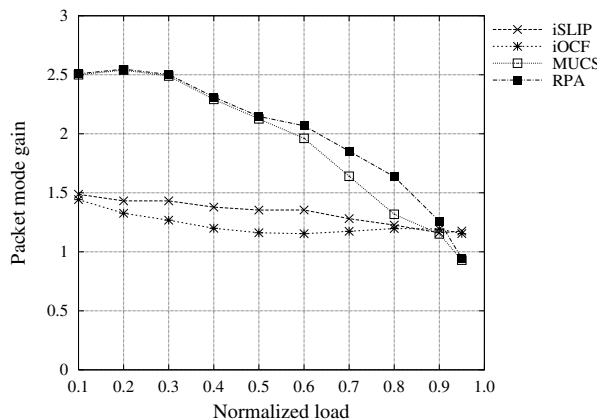


Fig. 9. Packet-mode gain for uniform packet length distribution

be observed in Fig. 8. Note that the delay increase for short packets can be detrimental to some applications (e.g., real-time applications using short packets).

E. Packet-mode gains

Next, we show some simulation results concerning the actual packet-mode gain values under different packet lengths distributions. Fig. 9 shows G for uniform distribution of packet lengths (with $C_v = 0.287$). Fig. 10 shows G for a bimodal distribution of packet lengths (with $C_v = 4.975$). Fig. 11 shows G for negative exponential distribution of packet lengths.

Table I compares the theoretical values with the experienced packet-mode gains for different scenarios, in the case in which iSLIP is adopted. We observe that the packet-mode gains obtained by simulation can be well estimated by the theoretical values, especially at low load. The proposed queuing model is thus capable of capturing the key aspects of the behavior of cell- and packet-mode iSLIP at low load.

Note also that iOCF in cell-mode shows very poor performance under bimodal packet length distribution for input load equal to 0.95. This is due mainly to the matching used in iOCF, which is not maximal; this drawback can be overcome by iterat-

ing the algorithm N times, as can be experienced by simulation.

VI. CONCLUSIONS

The paper focused on architectures and scheduling algorithms for IQ packet switches, comparing them with OQ switches in the case of variable-size data units.

Scheduling algorithms are required for the operation of all-non-purely-OQ architectures. We considered four previously proposed scheduling algorithms for the transfer of fixed-size data units, and proposed novel modifications of these scheduling algorithms to deal with variable-size packets, having in mind IP routers internally using ATM switching engines. These *packet-mode* scheduling algorithms require a negligible complexity increase with respect to cell scheduling, and can yield performance advantages over OQ architectures.

We analytically proved that no throughput limitations exist by operating a switch in packet mode, showed that relative delays depend on the traffic characteristics, and validated our analytical findings with simulation experiments. The results presented in this paper can influence the design of next-generation packet switches, since packet-mode scheduling in IQ architectures makes OQ architectures less attractive for the implemen-

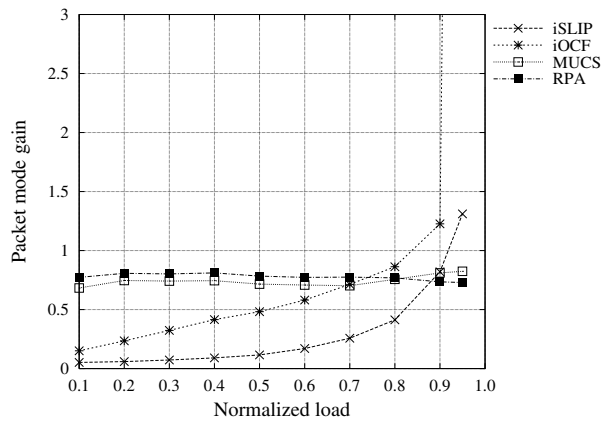


Fig. 10. Packet-mode gain for bimodal packet length distribution

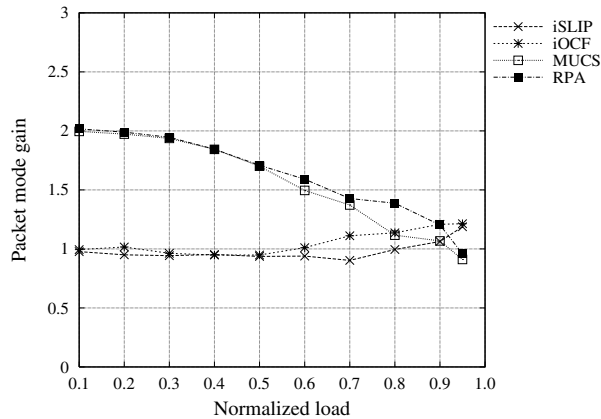


Fig. 11. Packet-mode gain for exponential packet length distribution

tation of high-performance IP routers.

REFERENCES

- [1] Anderson T., Owicki S., Saxe J., Thacker C., "High speed switch scheduling for local area networks", *ACM Trans. on Computer Systems*, vol. 11, n. 4, Nov. 1993, pp. 319-352
- [2] Parekh A.K., Gallager R.G., "A generalized processor sharing approach to flow control in integrated services networks - the single node case", *IEEE/ACM Trans. on Networking*, vol. 1, n. 3, June 1993, pp. 344-357
- [3] Demers A., Keshav S., Shenker S., "Analysis and simulation of a fair queueing algorithm", *ACM SIGCOMM'89*, Sept. 1989, pp. 3-12, Austin, TX
- [4] Golestani S., "A self-clocked fair queueing scheme for broadband applications", *IEEE INFOCOM'94*, Jun. 1994, pp. 636-646, Toronto, CA
- [5] Bennett J.C.R., Zhang H., "WF²Q: Worst-case fair weighted fair queueing", *IEEE INFOCOM'96*, Mar. 1996, pp. 120-128, San Francisco, CA
- [6] Karol M., Eng K., Obara H., "Improving the performance of input-queued ATM packets switches", *IEEE INFOCOM 92*, May 1992, pp. 110-115, Firenze, Italy
- [7] Tamir Y., Frazier G., "High performance multi-queue buffers for VLSI communication switches", *15th Ann. Symp. on Comp. Arch.*, June 1988, pp. 343-354, Washington DC
- [8] Tamir Y., Chi H. C., "Symmetric crossbar arbiters for VLSI communication switches", *IEEE Trans. on Parallel and Distributed Systems*, vol. 4, n. 1, pp. 13-27, Jan. 1993
- [9] McKeown N., "Scheduling algorithms for input-queued cell switches", *Ph.D. Thesis*, Un. of California at Berkeley, 1995
- [10] Siliadis D., Varma A., "Providing bandwidth guarantees in an input

TABLE I

PACKET-MODE GAINS FROM THEORY AND SIMULATION FOR iSLIP

C_v	G			
	by theory	by simulation for loads:		
		0.1	0.3	0.5
0.287	1.84	1.49	1.43	1.35
1.00	1.00	0.98	0.94	0.93
4.975	0.077	0.052	0.072	0.11

buffered crossbar switch", *IEEE INFOCOM'95*, vol. 3, 1995, pp. 960-8, Los Alamitos, CA

- [11] Ajmone Marsan M., Bianco A., Leonardi E., Milia L., "RPA: a flexible scheduling algorithm for input buffered switches", *IEEE Trans. on Communications*, vol. 47, n. 12, Dec. 1999, pp. 1921-33
- [12] Duan H., Lockwood J.W., Kang S.M., Will J.D., "A high performance OC12/OC48 queue design prototype for input buffered ATM switches", *IEEE INFOCOM'97*, vol. 1, 1997, pp.20-8, Los Alamitos, CA
- [13] McKeown N., Mekkittikul A., "A practical scheduling algorithm to achieve 100% throughput in input-queued switches", *IEEE INFOCOM'98*, vol. 2, 1998, pp. 792-9, New York, NY
- [14] McKeown N., Anderson T.E., "A quantitative comparison of scheduling algorithms for input-queued switches", *Computer Networks & ISDN Systems*, vol. 30, n. 24, Dec. 1998, pp. 2309-26
- [15] Ajmone Marsan M., Bianco A., Filippi E., Giaccone P., Leonardi E., Neri F., "On the behavior of input queuing switch architectures", *European Trans. on Telecommunications*, vol. 10, n. 2, Mar. 1999, pp. 111-124
- [16] "GFR MultiGigabit Routers", Product Overview, www.lucent.com, Apr. 2000
- [17] "Cisco 12000 Gigabit Switch Router", Product Overview, www.cisco.com, Apr. 2000
- [18] McKeown N., Izzard M., Mekkittikul A., Ellesick B., Horowitz M., "The Tiny Tera: a packet switch core", *IEEE Micro Magazine*, vol. 17, Feb. 1997, pp. 27-40
- [19] Hung A., Kesidis G., McKeown N., "ATM input-buffered switches with guaranteed-rate property", *IEEE ISCC'98*, July 1998, pp. 331-335, Athens, Greece
- [20] Partridge C., et al., "A 50-Gb/s IP router", *IEEE Trans. on Networking*, vol. 6, n. 3, June 1998, pp. 237-248
- [21] Karol M., Hluchyj M., Morgan S., "Input versus output queuing on a space division switch", *IEEE Trans. on Communications*, vol. 35, n. 12, Dec. 1987, pp. 1347-1356
- [22] Tarjan R.E., *Data structures and network algorithms*, Society for Industrial and Applied Mathematics, Pennsylvania, Nov. 1983
- [23] Even S., Kariv O., "An $O(n^{2.5})$ algorithm for maximum matching in general graphs", *16th Symp. on Foundations of Computer Science*, Un. of California at Berkeley, Oct. 1975, pp. 100-112
- [24] Hopcroft J.E., Karp R.M., "An $n^{2.5}$ algorithm for maximum matching in bipartite graphs", *Society for Industrial and Applied Mathematics J. Comput.*, vol. 2, 1973, pp. 225-231
- [25] McKeown N., "iSLIP: a scheduling algorithm for input-queued switches", *IEEE Trans. on Networking*, vol. 7, n. 2, Apr. 1999, pp. 188-201
- [26] Kumar P.R., Meyn S. P., "Stability of queueing networks and scheduling policies", *IEEE Trans. on Automatic Control*, vol. 40, n. 2, Febr. 1995, pp. 251-260
- [27] Wolff R.W., *Stochastic modeling and the theory of queues*, Prentice-Hall, NJ, 1989
- [28] McKeown N., Mekkittikul A., Anantharam V., Walrand J., "Achieving 100% throughput in an input-queued switch", *IEEE Trans. on Communications*, vol. 47, n. 8, Aug. 1999, pp. 1260-7
- [29] Kleinrock L., *Queueing systems - Volume 2: Computer applications*, John Wiley and Sons, 1976
- [30] Atkinson R., *RFC 1626: Default IP MTU for use over ATM AAL5*, May 1994