

Optimal Multicast Scheduling in Input-Queued Switches

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

Dipartimento di Elettronica – Politecnico di Torino – Italy

email:{ajmone, bianco, giaccone, leonardi, neri}@polito.it

Abstract— This paper focuses on multicast support in input-queued packet switches with internal multicast capabilities. Besides providing an overview of some alternative architectures and algorithms proposed in the literature, the paper brings two original contributions. First, multicast traffic admissibility conditions are defined, and theorems showing intrinsic performance losses of input-queued with respect to output-queued switch architectures are proved. Second, the optimal scheduling discipline in transferring multicast packets from switch inputs to switch outputs is defined. From the definition of the optimal multicast scheduling discipline, the formal characterization of the sustainable multicast traffic region naturally follows. Both results aim at a correct formal definition of the considered problem, in order to identify a sound starting point for the design of heuristics that approximate the optimal solution at a complexity compatible with available technologies.

I. INTRODUCTION

We consider input-queued packet switches or routers where arriving unicast and multicast fixed- or variable-size packets are fragmented into fixed-size cells and stored into buffers at input ports. Cells are transferred from input to output ports through a switching fabric with multicast capabilities, following a scheduling discipline that must avoid contentions (no more than one cell can be extracted from an input port in one time slot, and no more than one cell can be delivered to an output port in one time slot).

In input-queued switches the main issue concerning multicast transmissions is the capability of transferring a multicast cell from an input queue to different output ports. To solve this problem, several switch architectures have been proposed [1], which 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.

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 by the input port can reach any

number of output ports within one time slot. In this paper we refer to input-queued switches whose internal fabrics have intrinsic multicast capabilities.

The problem of scheduling multicast traffic was defined and modelled in [2], using a theory based on stochastic ordering and majorization. In that work, the optimal scheduling discipline was fully characterized for a switch with two and three input ports, based on a queueing structure with only one FIFO queue for each input. No results about the maximum achievable throughput were provided.

Several theoretical studies [3], [4], [5], [6] have appeared, that investigate the maximum throughput achievable when arrivals 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 for uniform traffic is always less than one, and that it depends on the multicast traffic distribution.

In this paper we tackle some fundamental problems concerning the transfer of multicast traffic within input-queued switching architectures.

In Section II we describe architectural alternatives for packet switches with multicast support, and we define the reference architecture for this paper. We focus on three elements: the cell flows entering the switch, the input-queueing structure, and the scheduling discipline. Section III discusses traffic sustainability conditions, i.e., the constraints on input traffic to guarantee finite queue lengths in the switch. Examples of traffic patterns for which input-queued switches exhibit intrinsic performance losses with respect to output-queued switch architectures are provided. Section IV formally defines the optimal multicast packet scheduling discipline and the capacity region. For the sake of readability, analytical details were moved to the Appendix. Finally, Section V concludes the paper.

II. MULTICASTING IN INPUT-QUEUED SWITCHES

We introduce in this section some basic definitions on multicast switching, and define the switch architecture considered in the paper. Unless otherwise specified, we

This work was supported by the Italian Ministry for University and Scientific Research.

TABLE I
EXAMPLE OF A 2-COMPLEX REQUEST SET FOR A 2×6 SWITCH

ID	Input	Outputs
1	0	{0, 1, 2}
2	0	{0, 3, 5}
3	1	{1, 4, 5}
4	1	{2, 3, 4}

refer to switches with N input and N output ports (numbered from 0 to $N - 1$), in which all input and output lines run at the same data rate. Since we only consider fixed-size data units, we use the terms “packet” and “cell” interchangeably throughout the paper.

A. Multicast Traffic Patterns

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 *fanout* [5], [6] 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 equal to one, and its fanout destination is the only output port to which the cell is destined.

We introduce a novel class of traffic patterns, which will be useful for our theoretical considerations.

We call *request set* R a set of multicast packets to be transferred from inputs to outputs.

Definition 1: A request set R comprising kL packets is said **k -complex**, with¹ $k \in \mathbb{N}$, $k > 1$, $L \in \mathbb{N}$, and $L > 1$, if:

- k packets are stored at any given input;
- no more than k packets are directed to a given output;
- for each sub-set of k packets at least a destination can be found to which all the packets in the subset are directed.

Table I reports an example of a 2-complex request set for a 2×6 switch; ID is a packet identifier.

A traffic pattern consisting of a sequence of k -complex request sets is said to be a k -complex traffic pattern.

The average amount of traffic at each input (output) is called input (output) load. In this paper we normalize input (output) loads to line rates: a load equal to 1 means a fully utilized input (output) line. The traffic at the input of a switch is called *admissible* if no input load is larger than 1, and no output load is larger than 1. An input traffic is called *sustainable* if it can be transferred through the switch. We shall see in the paper that an admissible traffic is also sustainable in output-queued switches, not in input-queued switches.

A normalized input load $\rho \in [0, 1]$ is generated by a k -complex traffic pattern by letting one packet among those

¹In 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 non negative real numbers.

defined in the request set arrive at each active input with probability ρ in every time slot. One of the k packets assigned to an active input by the k -complex request set is selected with a uniform probability and queued at the corresponding input. If ρ tends to 1, the average number of packets that are directed to each output during k consecutive time slots is equal to k . Thus, if ρ is the input load, ρ equals the output load. If ρ is a sustainable load (and no losses occur in the switch) the throughput is also ρ for each output.

B. The Input Queue Structure

It is known that under multicast traffic, buffer space may be used more efficiently by input-queued (IQ) switches rather than output-queued (OQ) switches, in the sense that any multicast cell currently in the switch is stored only in one buffer position.

Several different ways of organizing the input queue system can be envisaged. We suppose to use at each input a set of FIFO queues, and that the algorithm that computes the scheduling of cell transfers is supposed to scan only the cells at the heads of the input queues.

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 [7] head-of-the-line (HOL) blocking effect, which limits the maximum throughput achievable by IQ switches. In the case of unicast traffic only, a common approach for avoiding HOL blocking in IQ switches consists of using at each input separate queues for each output; this 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 consider a queue architecture with $N(2^N - 1)$ FIFO queues; $(2^N - 1)$ queues are present at each input, one for each of the $(2^N - 1)$ possible fanout configurations. Given a specific switch output, $N2^{N-1}$ input queues store packets directed to it. This structure completely avoids HOL blocking, but requires a huge number of queues. Although it can be implemented by managing logical queues inside a Random Access Memory (RAM), like in a flow queueing structure, it can be impractical for real implementations of large switches. We only consider it as a best case, ideal situation, which permits our theoretical considerations. Note that, by avoiding any form of HOL blocking, the system performance studied with this ideal queueing structure is an upper bound for the throughput. To simplify our theoretical considerations, we also assume infinite queue lengths.

C. The Scheduling Discipline

At each time slot, cells stored in input queues contend for access to the switching fabric to reach output ports.

The decision about which cells can be transferred is made by the scheduling discipline.

The fact that multicast cells have multiple destinations implies that some scheduling disciplines may elect to transfer the multicast cell only when all destinations are reachable, in just one time slot, while others may elect to transfer the cell in several time slots, reaching exclusive and exhaustive subsets of destinations. 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* [5], [9] is defined as the set of fanout destinations that have not yet been reached after a multicast cell is transferred towards output ports.

Although a hybrid approach can be adopted, one of the two following basic strategies is normally used by the scheduling discipline:

- **No fanout splitting** – Any multicast cell is transferred through the switching fabric only once, and only if all fanout destinations can be reached in the same time slot. If any of the fanout destinations cannot be reached because the multicast cell loses contention for an output port, the cell cannot be transferred, and it will contend again in a future time slot (normally the next one). This strategy is non-work-conserving, since it may happen that no cell is delivered to an available output because of contention. This discipline may favor cells with small fanout.
- **Fanout splitting** [5] – Multicast cells may be delivered to output ports over a number of time slots. Only those fanout destinations that could not be reached in previous time slots are considered in the next time slot.

Note that no fanout splitting scheduling disciplines imply that a total service is always adopted, whereas multicast cells can receive either total or partial service if a fanout splitting discipline is used.

In the case of fanout splitting, every partial service causes an increase of the input load, similarly to what happens in the case of the copy network approach, leading to performance penalties. This effect is missing if a no fanout splitting discipline is adopted, but in this case another form of performance penalty is introduced, due to the non work-conserving service. A good scheduling discipline should therefore find the best compromise between these two major performance impairment, that are further discussed in the next section.

Multicast cells that receive a partial service must remain in input queues. Because of the particular queue structure we envision, when a cell receives partial service, it must be moved to a different queue after being served, specifically to the queue corresponding to the remaining residue. In the worst case, two write operations (one for the new incoming cell, the other for the residue) and one read operation have to be supported in the same queue. This re-enqueueing process can produce out-of-sequence

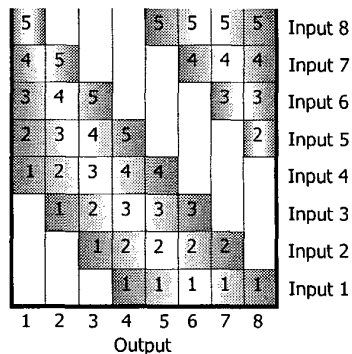


Fig. 1. Example scenario leading to poor throughput with no fanout splitting policies

cell delivery, since residues left by packets belonging to the same flow can change and be enqueued into different queues. This is probably unacceptable for practical implementations, but it is instrumental to study the best throughput available in an IQ switch.

III. TRAFFIC SUSTAINABILITY CONDITIONS

As it was shown in [5], [6], [12], the maximum throughput that can be obtained with *no fanout splitting* policies (which are non-work-conserving) is lower than with *fanout splitting* policies (which are work-conserving); we now provide a trivial example of this fact.

Consider a deterministic traffic pattern such that all cells arriving at any input i of a $N \times N$ switch are multicast with fanout F , resulting from the fanout set B_i :

$$B_i = \{i, |i + 1|_N, \dots, |i + F - 1|_N\}$$

where $|x|_N$ is the remainder of the division of x by N , and F is the smallest integer larger than $(N + 1)/2$. Each multicast cell arriving at input i can contend with all other $N - 1$ multicast cells arriving at other inputs, because of the chosen fanout sets. Figure 1 shows an example of the considered traffic pattern in the case of $N = 8$ and $F = 5$. Each multicast cell is depicted as a set of F horizontal blocks. Each block corresponds to a single fanout destination of the cell. A block in row i and column j is generated from input i and it is directed to output j . The number inside a block is the time slot when the block is transmitted to the outputs if a fanout splitting policy is adopted. To schedule all the blocks, F time slots are needed. When a no fanout splitting policy is adopted, N time slots are necessary to deliver all the blocks. The maximum admissible input load λ is equal to $1/F$ for fanout splitting policies and $1/N$ for no fanout splitting policies, with $\frac{1}{F} \approx \frac{2}{N}$. As discussed in Section IV-B, for this particular scenario, $1/F$ is also the maximum throughput achievable in an OQ switch.

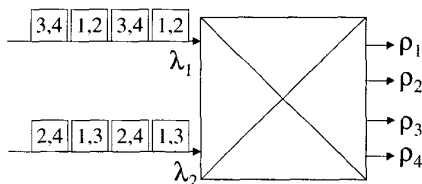


Fig. 2. Example traffic pattern leading to poor throughput with both fanout and no fanout splitting policies

With multicast traffic, the constraints for traffic admissibility are well-defined only for OQ switch architectures: no input load and no output load must be larger than 1. In the case of IQ switches, instead, while the best scheduling discipline for unicast traffic can sustain the same load of an OQ switch, it is easy to find a traffic pattern for which the best fanout splitting algorithm cannot reach the maximum throughput. When fanout splitting is allowed, 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 produces additional load, was observed in [6].

The example shown in Figure 2 gives an indication of the problems that can arise with unfortunate traffic patterns. Numbers inside packets show their fanout destinations. Using an OQ switch architecture, it is possible to sustain the considered traffic pattern since the load offered to each input and to each output is smaller than one. Instead, with IQ switches and the best fanout splitting strategy, $3/2$ cells can be scheduled in each time slot in the whole switch. This means that, for each input, $3/4$ cells are scheduled in each time slot, so that each cell requires $4/3$ time slots to be scheduled, and the bandwidth reduction factor is $\alpha \approx 1.33$.

Thus, with the setup shown in Figure 2, while an OQ switch can sustain the considered traffic pattern until $\lambda_0 < 1$ and $\lambda_1 < 1$, IQ switches must impose $\lambda_0 < 0.75$ and $\lambda_1 < 0.75$.

In summary, in an IQ switch:

- a replication scheme replicates each cell and increases the load offered to the switching fabric to a level that is not sustainable;
- a no fanout splitting policy wastes the bandwidth available in the switching fabric since it is non-work-conserving;
- a fanout splitting policy is work-conserving but each splitting increases the effective input load to the fabric, since one cell needs several time slots to be completely scheduled.

We can now prove two original results.

Theorem 1: With specific admissible input traffic patterns, that lead to throughput one in OQ switches, IQ switches using a no fanout splitting policy provide a throughput that can drop to zero if the number of ports is very large.

Proof: We consider a switch with L active inputs receiving a 2-complex (see Definition 1) traffic pattern. An OQ architecture can transfer sL packets every s time slots and can sustain an output load almost equal but strictly less than one, given that the input load is admissible. Conversely, in an IQ switch using a no fanout splitting policy, at most one cell can be scheduled in each time slot, that is, at most s packets are transferred every s time slots. Thus, the throughput for IQ with no fanout splitting is $1/L$ of the throughput for OQ, and drops to zero when L grows indefinitely. ■

Theorem 2: With specific admissible input traffic patterns, the maximum sustainable throughput for an IQ switch using a fanout splitting policy is ≤ 0.5 .

Proof: We show a case in which the maximum load per input should be less or equal to 0.5 to keep the system stable. We consider a large size switch loaded with a 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 for 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 transmitted as a whole. The complete transmission of two packets in the same time slot would, indeed, lead to a conflict on the output port to which both packets are directed. As a consequence, most of the packets must be split in at least two parts by any work-conserving scheduler. In each time slot at most one packet can be completely transmitted, and no more than $k - 1$ packets can be partially transmitted. Thus the maximum number of packets transmitted in each slot cannot exceed $T = 1 + (k - 1)(2)$ where the transmission of a single part of a packet split in two counts as half packet transmission, since two time slots are required to transmit the full packet. The effective service rate per input port is: $\mu = T/k = (k + 1)/(2k)$. Since $\mu > 0.5$ and $\lim_{k \rightarrow \infty} \mu = 0.5^+$, the stability condition requires that $\lambda \leq 0.5$. ■

Note that Theorem 2 can be referred also to switching architectures with internal speedup: the minimum speedup required to achieve 100% throughput under any admissible traffic pattern is no less than two for a large size switch.

IV. OPTIMAL SCHEDULING

In this section we define the optimal scheduling discipline for an IQ switch. Our methodology is based on the approach used in [13], [14], [15], [16]. We first introduce our notation and some useful relations.

A. Switch Description

We define the *fanout index* as an integer between 1 and $(2^N - 1)$, which identifies a specific fanout set with the following rule: output j ($j = 0, 1, \dots, N-1$) is included in the fanout set described by fanout index f when the j -th bit is equal to 1 in the binary representation of f on N bits. In other words, the fanout set can be represented as a binary string of N bits where the j -th bit is equal to 1 when the j -th output is included in the fanout set. Each queue at an input port can be univocally identified by its fanout index because of the particular queue architecture.

Let $Q_n^{i,f}$ be the number of cells queued at input i with fanout index equal to f measured at discrete time n ($n \geq 0$). We define X_n^i as the row vector of queues lengths corresponding to input i . i.e., $X_n^i = [Q_n^{i,1}, Q_n^{i,2}, \dots, Q_n^{i,2^N-1}]$. The row vector $X_n = [X_n^0, X_n^1, \dots, X_n^{N-1}]$ describes the lengths of all $N(2^N - 1)$ queues, and it is considered globally as a row vector of $N(2^N - 1)$ elements; each element corresponds to a particular queue in the input queue structure.

Let $A_n = [a_n^k]$ be a row vector composed by $N(2^N - 1)$ elements; the k -th element a_n^k corresponds to the number of cells arrived at the corresponding queue in X_n in time interval $(n, n + 1]$. A_n is defined as the **arrival vector** and is a realization of stochastic process A . $D_n = [d_n^k]$ is the **departure vector** defined in a way similar to A_n ; d_n^k is the number of cells leaving the corresponding queue in time interval $(n, n + 1]$. We assume $a_n^k \in \{0, 1\}$ and $d_n^k \in \{-1, 0, 1\}$; the -1 value of d_n^k is now explained. When fanout splitting is allowed, only a part of the fanout set is switched, and the residue is buffered in the queue corresponding to the remaining fanout destinations. If a cell is transmitted from queue k_1 and it is re-queued into queue k_2 , then $d_n^{k_1} = 1$ and $d_n^{k_2} = -1$.

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 $\Psi_I(i)$ be the set of $(2^N - 1)$ queues corresponding to input i ; let $\Psi_O(j)$ be the set of $N(2^N - 1)$ queues corresponding to output j . Let w_i^I be a binary row vector of size $N(2^N - 1)$, whose k -th element is equal to 1 iff $k \in \Psi_I(i)$. Let w_j^O be a binary row vector of size $N(2^N - 1)$, whose k -th element is equal to 1 iff $k \in \Psi_O(j)$. Let $w_{ij}^{I,O}$ be a binary row vector of size $N(2^N - 1)$, whose k -th element is equal to 1 iff $k \in \Psi_I(i) \cap \Psi_O(j)$. Note that $w_{ij}^{I,O} = w_i^I \wedge w_j^O$, where \wedge denotes logical and. If V is a vector, let $u(V)$ be a vectorial operator which outputs

a vector U (with the same size of V) whose k -th 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 equations:

$$0 \leq w_i^I A_n^T \leq 1 \quad (1)$$

$$0 \leq w_i^I u(D_n^T) \leq 1 \quad (2)$$

$$0 \leq w_i^I u(-D_n^T) \leq 1 \quad (3)$$

$$0 \leq w_j^O u(D_n^T) \leq 1 \quad (4)$$

$$0 \leq w_{ij}^{I,O} D_n^T \leq 1 \quad (5)$$

for $1 \leq i, j \leq N$ and $n \geq 0$. (1) means that at most one multicast cell can arrive at each input during a time slot; (2) means that at most one cell can be forwarded from each input; at most one cell can be re-queued into a queue, according to (3); (4) means that at most one cell is sent to an output for each time slot; by (5), each packet which receives a partial service is moved to a queue corresponding to a subset of the initial fanout.

B. Admissible Region

The arrival vector is a realization of a stochastic process described by the row vector Λ , which is defined as the expected value of the arrival process vector A_n : $\Lambda \triangleq E[A_n]$.

We define the input traffic as admissible when neither input nor output ports are overloaded. With our notation, avoiding input overload can be specified as

$$0 \leq w_i^I \Lambda^T < 1 \quad \text{for } i = 0, 1, \dots, N-1 \quad (6)$$

To avoid output overloading, we require instead that:

$$0 \leq w_j^O \Lambda^T < 1 \quad \text{for } j = 0, 1, \dots, N-1 \quad (7)$$

Note that (6) and (7) are only necessary conditions for the stability of an IQ switch, while they guarantee stability to an OQ switch. The set of Λ satisfying (6) and (7) is called admissible region.

C. Throughput Definition

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

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

D. Optimal Policy and Capacity Region

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

Referring to the stochastic version of Lyapunov stability [13], the maximum throughput can be obtained by

solving the following optimization problem at each discrete time n :

$$D_n^* = \arg\{\max_{D_n}\{D_n X_n^T\}\} \quad (8)$$

subject to constraints (2), (3), (4) and (5). D_n^* defines the optimal scheduling choice during time slot n and it is called **max-scalar discipline**. The proof of the optimality of the max-scalar discipline is reported in Appendix A.

For a given X_n , the traffic patterns that satisfy the following relation keep stable the queue lengths

$$\Lambda X_n^T \leq D_n^* X_n^T \quad (9)$$

hence they are sustainable. As shown in the Appendix, to satisfy (9), the average arrival vector Λ must be inside the convex hull produced by all possible departure vectors $D^{(k)}$ in the IQ switch. The solution can be obtained by solving

$$\Lambda = \sum_k \alpha_k D^{(k)} \quad (10)$$

$$\text{subject to } \begin{cases} \sum_k \alpha_k = 1 \\ 0 \leq \alpha_k \leq 1 \end{cases} \quad \forall k \quad (11)$$

The set of all Λ which permit solutions to the above problem defines the **capacity region** for multicast traffic in an IQ switch. The capacity region contains all average arrival vectors for which the queues of our input queueing structure remain finite. The examples and the theorems shown in Section III prove that the capacity region is strictly internal to the admissible region. For OQ switches the two regions coincide.

By solving the problem defined by relations (10) and (11), it is possible to find the optimal discipline to schedule multicast traffic in an IQ switch. The solution can be found by solving a linear programming problem, but numerical computation is practically limited by the huge number of possible departure vectors $D^{(k)}$, which corresponds to the number of switching configurations.

V. CONCLUSIONS

The paper has proposed a formalization of the problem of supporting multicast traffic in input-queued packet switches with internal multicast capabilities. Traffic admissibility conditions were defined, and two theorems showing intrinsic performance losses of input-queued switches with respect to output-queued switch architectures were proven. An optimal scheduling discipline, called "max-scalar", to transfer packets from switch inputs to switch outputs 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 we obtained the formal characterization of the capacity region, i.e., the set of input loads that an ideal IQ switch is able to transfer to output ports.

The results presented in the paper are based on an idealized switch model, and aim at a correct formal definition of the considered problem, to identify a sound starting point for the design of heuristics that approximate the optimal solution at a complexity compatible with implementation using available technologies.

REFERENCES

- [1] Guo M., Chang R., "Multicast ATM switches: survey and performance evaluation", *Computer Communication Review*, vol. 28, no. 2, pp. 98-131, Apr. 1998
- [2] Liu Z., Righter R., "Scheduling multicast input-queued switches", *Journal of Scheduling*, John Wiley & Sons, May 1999
- [3] Chen X., Lambadaris I., Hayes J., "A general unified model for performance analysis of multicast switching", *IEEE GLOBECOM'92*, New York, NY, USA, vol. 3, pp. 1498-502, 1992
- [4] Hayes J.F., Breault R., Mehmet-Ali M.K., "Performance analysis of a multicast switch", *IEEE Transactions on Communications*, vol. 39, no. 4, pp. 581-587, Apr. 1991
- [5] Hui J., Renner T., "Queueing strategies for multicast packet switching", *IEEE GLOBECOM'90*, pp. 1431-1437, 1990
- [6] Kim C.K., Lee T.T., "Performance of call splitting algorithms for multicast traffic", *IEEE INFOCOM'90*, pp. 348-356, 1990
- [7] Hluchyj M.G., Karol M.J., Morgan S., "Input versus output queueing on a space division switch", *IEEE Transactions on Communications*, vol. 35, n. 12, pp. 1347-1356, Dec. 1987
- [8] Hluchyj M.G., Karol M.J., "Queueing in high-performance packet switching", *IEEE Journal on Selected Areas in Communications*, vol. 6, pp. 1587-1597, Dec. 1988
- [9] McKeown N., Prabhakar B., "Scheduling multicast cells in an input-queued switch", *INFOCOM'96*, vol. 1, pp. 261-278, San Francisco, Mar. 1996
- [10] Yeung K.L., Au-Yeung K.F., Ping L., "Efficient time slot assignments for TDM multicast switching systems", *1997 IEEE International Conference on Communications, ICC '97*, IEEE, New York, NY, USA, vol. 3, pp. 1366-70, 1997
- [11] Prabhakar B., McKeown N., Ahuja R., "Multicast scheduling for input-queued switches", *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 5, pp. 855-66, Jun. 1997
- [12] Kim C.K., Lee T.T., "Call scheduling algorithm in multicast switching systems", *IEEE Transactions on Communications*, vol. 40, n. 3, pp. 625-635, Mar. 1992
- [13] Ajmone Marsan M., Leonardi E., Mellia M., Neri F., "On the stability of input-buffered cell switches with speed-up", *IEEE INFOCOM 2000*, Tel-Aviv, Israel, Mar. 2000
- [14] Kumar P.R., Meyn S.P., "Stability of queueing networks and scheduling policies", *IEEE Transactions on Automatic Control*, vol. 40, n. 2, pp. 251-260, Feb. 1995
- [15] McKeown N., Anantharam V., Walrand J., "Achieving 100% throughput in an input-queued switch", *IEEE INFOCOM'96*, vol. 1, pp. 296-302, San Francisco, Mar. 1996
- [16] Tassiulas L., Ephremides A., "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks", *IEEE Transactions on Automatic Control*, vol. 37, n. 12, pp. 1936-1948, Dec. 1992

APPENDIX

I. 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 2: A system of queues achieves **100% throughput** if

$$\lim_{n \rightarrow \infty} \frac{X_n}{n} = \lim_{n \rightarrow \infty} \frac{1}{n} \sum_{i=0}^{n-1} (A_i - D_i) = 0 \quad \text{w. p. 1}$$

Definition 3: A system of queues is said to be **strongly stable** if

$$\limsup_{n \rightarrow \infty} E[\|X_n\|] < \infty$$

Theorem 3: 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 $V(X_n) = X_n W X_n^T$ (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 [14], [16].

Definition 4: Let $CH\{D^{(k)}\}_k$ represent the **convex hull** generated by all possible admissible departure vectors $D^{(k)}$.

Theorem 4: A necessary condition to achieve 100% throughput in an IQ switch is that the input traffic process, described by a sequence of arrival vectors A_n and satisfying the strong law of large numbers, hence stationary and ergodic, satisfies $E[A_n] \in CH\{D^{(k)}\}_k$.

Proof: Suppose that the switch achieves 100% throughput, then:

$$\lim_{n \rightarrow \infty} \frac{X_n}{n} = 0 \quad \text{w. p. 1}$$

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

$$\begin{aligned} \lim_{n \rightarrow \infty} \frac{X_n}{n} &= \lim_{n \rightarrow \infty} \frac{\sum_{i=0}^{n-1} (A_i - D_i)}{n} = \\ &= \lim_{n \rightarrow \infty} \frac{\sum_{i=0}^{n-1} A_i}{n} - \lim_{n \rightarrow \infty} \frac{\sum_{i=0}^{n-1} D_i}{n} = \\ &= E[A] - \lim_{n \rightarrow \infty} \frac{\sum_{i=0}^{n-1} D_i}{n} = 0 \quad \text{w. p. 1} \end{aligned}$$

A is the stationary and ergodic arrival process and A_n is 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

$$E[A] = \lim_{n \rightarrow \infty} \frac{\sum_{i=0}^{n-1} D_i}{n} \quad \text{w. p. 1}$$

This means that, for some realizations of D_i , $\forall \epsilon > 0$, there exists 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 $\frac{\sum_{i=0}^{n-1} D_i}{n}$, 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 5: 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 $X_n \neq 0$,

$$\frac{E[A_n]X_n^T}{\|X_n\|} < \frac{\hat{A}_n X_n^T}{\|X_n\|} = \frac{E[A_n]X_n^T}{\|X_n\|} \alpha \leq \frac{D_n^* X_n^T}{\|X_n\|}$$

where \hat{A} is the vector parallel to $E[A_n]$ on the border of $CH\{D^{(k)}\}_k$, D_n^* is defined as in (8), and $\alpha = \frac{\|\hat{A}_n\|}{\|E[A_n]\|} > 1$. We can thus write Since $\hat{A}_n = \alpha E[A_n] = E[A_n] - (\alpha - 1)E[A_n]$, we write:

$$\frac{(\hat{A}_n - D_n^*)X_n^T}{\|X_n\|} = \frac{(E[A_n] - (\alpha - 1)E[A_n] - D_n^*)X_n^T}{\|X_n\|} \leq 0$$

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

$$\begin{aligned} \lim_{\|X_n\| \rightarrow \infty} \frac{(E[A_n] - D_n^*)X_n^T}{\|X_n\|} &\leq (1 - \alpha) \frac{E[A_n]X_n^T}{\|X_n\|} \\ &\leq (1 - \alpha)r_{\min} < 0 \end{aligned}$$

(note that $(1 - \alpha)$ is a negative non-null quantity). By using the quadratic Lyapunov function associated with the identity matrix [13], [15], i.e., $V(X_n) = X_n X_n^T$, and assuming the max-scalar scheduling discipline, we have

$$\begin{aligned} E[V(X_{n+1}) - V(X_n) | X_n] &= \\ E[X_{n+1}X_{n+1}^T - X_n X_n^T | X_n] &= \\ E[(X_n + A_n - D_n^*)(X_n + A_n - D_n^* D_n^*)^T - X_n X_n^T] &= \\ E[2(A_n - D_n^*)X_n^T - (A_n - D_n^*)(A_n - D_n^*)^T] & \end{aligned}$$

Being $(A_n - D_n^*)(A_n - D_n^*)^T$ negligible for $\|X_n\| \rightarrow \infty$, if we choose $\epsilon = -(1 - \alpha)r_{\min}/2$, we observe that there exists $B > 0$ and $\epsilon > 0$ such that, $\forall X_n : \|X_n\| > B$,

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

If in addition $E[A_n A_n^T] < \infty$, then $E[V(X_{n+1}) | X_n] < \infty$, $\forall X_n$; since the conditions of Theorem 3 hold, the system is strongly stable. ■

Since the strong stability of a system also implies 100% throughput, by combining the previous results it follows that:

Corollary 1: Under any i.i.d. sequence A_n of arrivals such that $E[A_n A_n^T] < \infty$, 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.