Multi-hop scheduling for optical switches with large reconfiguration overhead

Multi-hop scheduling for optical switches with large reconfiguration overhead

Andrea Bianco, Paolo Giaccone, Emilio Leonardi, Fabio Neri, Paola Rosa Brusin
Dipartimento di Elettronica, Politecnico di Torino, Italy
e-mail: {bianco,giaccone,leonardi,neri,brusin}@mail.tlc.polito.it

Abstract— **One of the limiting factors that prevents a widespread use of optical switching fabrics in high speed routers is their large reconfiguration overhead. Indeed, due to technological constraints, changing configuration in an optical switching fabric implies that the communication between input and output ports is established only after a given reconfiguration time. This reconfiguration time may be significantly larger than the packet transmission time, and cannot be considered negligible as in electronic switching fabrics.**

Recent works have addressed the problem of scheduling incoming traffic transmissions across optical fabrics, by extending the traditional approaches conceived for electronic switches. However, for very large switch size (1000's of ports) and high speed links (10's of Gbps), performance in terms of average delays may be unacceptable.

We propose a different approach, based on *multi-hop* scheduling, in which packets are relayed to the output port by a number of intermediate hops, i.e. successive transmissions through intermediate ports. Our approach achieves much smaller delays than those of traditional single-hop scheduling for low to medium loads, and comparable delays for high loads. We believe that multi-hop scheduling is a very promising scheme to achieve good performance when using optical switching fabrics in high-speed routers.

I. Introduction

All-optical switches are considered a very appealing solution for the design of ultra-high speed networks, the main advantage being the avoidance of optical-to-electronic conversions, which is a technological issue limiting the performance in current switches to hundreds of Gbps. Unfortunately, all-optical switches are practically infeasible for the lack of simple "optical memories" able to mimic the buffers used in electronic switches to solve temporary congestion.

Hybrid optical/electronic switching architectures are today the most promising approach to design routers able to reach aggregate bandwidths up to 100 Tbps [9]. The switching fabric we consider is fully optical and is typically located in a different rack with respect to the switch line-cards. In these architectures, packets arrive at the router through optical links, and, after the optical/electronic conversion, they are processed and buffered in the line-card; after an electronic/optical conversion, packets are sent over optical fibers to the optical switching fabric. Note that the use of optical switching fabrics may be convenient also to reduce power consumption, since in optics power consumption is largely independent from the transmission rate.

This work was supported by the Italian Ministry for Education, University and Research, within the EURO project.

Optical switching fabrics may be based on several different technologies, such as MEMS [5], bubble switches [6], broadcast and select networks with tunable devices [4], etc. However, most of these technologies share a common feature, i.e., whenever the switching fabric configuration (input/output ports connections) is changed, a *reconfiguration time* is required before communication takes place; note that at least the ports involved in such reconfiguration must refrain from transmission, but often all the ports of the switch are blocked until the reconfiguration time expires, and we make this assumption in this paper. The reconfiguration time, due to technological constraints like mechanical inertial effects in MEMS, or tuning times of tunable devices in broadcast and select networks, is usually not negligible with respect to the packet transmission times (which are in the order of few ns at very high line rates), and can adversely affect switch performance.

As a consequence, the scheduling algorithm, whose task is to select the switching configuration of the optical device, should take into consideration reconfiguration time constraints so as to minimize the number of reconfigurations required to efficiently transfer a given traffic pattern. To the best of our knowledge, only few works have been proposed that consider the additional constraints due to reconfiguration times when defining the scheduling problem (see [7], [8], [10], [13]). All these works assume that, when input i is connected to output j, only packets stored at input port i and destined to output port j can be transferred through the switching fabric, i.e. all the packets cross the switching fabric only once. In other words, when N packets are present at one input and destined to different N outputs, at least N switching fabric reconfigurations are required to allow the full connectivity between all inputs and outputs to be obtained, and to transfer N packets in sequence.

Scheduling algorithms must carefully balance two main performance objectives: throughput and delay. This balancement becomes fundamental when reconfiguration times are not negligible. Indeed, to obtain high throughput, the scheduling should keep for long times the same switching configuration, so as to reduce the negative effect of inactivity periods due to the reconfiguration overhead; however, low delays imply to change quickly the switching configuration, so as to allow the full connectivity between all ports to be obtained in a short time interval.

Consider the following scenario, with an optical switch with $N = 1024$ ports based on MEMS technology, with a reconfiguration time $R = 1$ ms; assume that the link speed is 10 Gbps,

0-7803-8375-3/04/$20.00 © 2004 IEEE

and that internally the switch operates on fixed-size data-unit of 128 bits (a convenient format for high speed electronic memories [9]); thus, the data-unit transmission time is $T = 12.8$ ns. Assume that, on an empty switch, N packets arrive, each at a different input, all destined for the same output. Even when keeping each switching matrix configuration for the minimum time required to transmit a single data-unit, thus obtaining a very low throughput efficiency, the k-th packet will be transferred at time $k(R + T)$ after arrival, because of the k reconfigurations needed before the connectivity to the desired output is provided to the input where packets are waiting. Hence, the worst delay for the N packets is $N(R+T) \approx NR \approx 1s$, which is obviously unacceptable for any realistic application; thus, if we do not take a different approach, this result may compromise the hopes towards the use of optical devices in routers in the future.

To overcome this problem, we propose in this paper to exploit a *multi-hop* approach, previously studied in the context of WDM/TDM networks [1]. The main idea of multi-hop scheduling, better described in Section III-A, is to configure the switching matrix once in a while, on a time scale significantly larger than packet transmission time, and to re-circulate packets among ports, i.e. a packet at input port s may reach its destination port d via successive transmissions through one (or more) intermediate ports. Thus, we exploit the fact that input port i and output port i reside always in the same line-card, and a packet arrived at output i can be reconsidered for retransmission across the switching fabric at negligible extra cost. Note that this architectural assumption is fairly common; e.g., see the single-stage switch described in [9]. In the same scenario previously considered, the worst case delay for a multi-hop approach, is simply $R + NT \approx R = 1$ ms, a much smaller value than the one obtained with the traditional single-hop approach; this delay can be acceptable for practical implementations.

Clearly, sending packets in multi-hop increases the overall load of the switching fabric; we show in this paper that we can deal with this issue, and that significant benefits in terms of delay can be obtained, especially for moderate to medium switch loads.

II. System Model

The model of the optical switch we use in this paper is similar to the single-stage architecture presented in [9]; however, we assume that a single optical switching fabric is available. This switching fabric behaves as a buffer-less crossbar. We also assume that input and output interfaces of a single port reside on the same line-card.

We consider a switch with N ports, each running at the same line rate r; all the packets arriving at the same input port and directed to the same output port belong to the same *flow*. Input queues are used to solve contentions among packets contending for the same output and are organized according to the Virtual Output Queueing (VOQ) buffering scheme, with one FIFO queue for each flow, to achieve high throughput [2]. Some extensions to this architecture will be considered and better described later, when the multi-hop approach is defined in detail.

We denote with λ_{ij} the average arrival rate at the queue at input i storing packets directed to output j; rates are normalized to the link rate. We consider only admissible traffic, i.e. not overloading neither inputs nor outputs:

$$\sum_{i=1}^{N} \lambda_{ij} < 1 \quad \forall j \quad \text{and} \quad \sum_{j=1}^{N} \lambda_{ij} < 1 \quad \forall i$$

Variable-size packets, when received, are segmented into fixed-size packets, which are individually switched to their destination port, where the original packet is reassembled. Hence, we assume that time is slotted, with the minimum time slot T corresponding to the duration of the fixed-size packet when transmitted at the line rate r.

III. Scheduling Algorithms

Traditional approaches to control optical devices [8], [10], [13] are always based on the *single-hop* idea, that is, each packet is transferred across the switching fabric once. Under this assumption, the scheduling problem at each time slot can be reduced to the selection of a matching, i.e. a set of input-output port pairs that are connected through the switching fabric, with the constraints that, at any given time, at most one input is connected to each output, and at most one output is connected to each input. Note that, in the single-hop approach, the choice of a matching defines univocally the set of VOQs from which packets are dequeued before entering the switching fabric.

When the switching configuration changes from one time slot to another, some input-output connections become unavailable for a certain period of time (named *reconfiguration time*) because of physical constraints of the optical devices. Thus, one *scheduling cycle* is composed by a fixed reconfiguration time of R time slots and by a fixed *persistence time* of P time slots, in which the same matching among ports is adopted to control the switching fabric.

With the assumptions above, the maximum achievable throughput for uniform traffic is simply $P/(P + R)$. Note that R depends on the reconfiguration time (given by the technological constraints of the adopted optical devices) and on the time slot duration T, which depends only on the internal packet size and on the transmission speed. Hence, the scheduler can adjust the value of P to achieve the required throughput: almost 100% throughput can be achieved for $P \gg R$. However, the experienced delays can be very large, since they increase linearly with the number of nodes and with the persistence time.

If the traffic matrix $\Lambda = [\lambda_{ij}]$ is known, the single-hop scheduling may choose to decompose Λ in a sequence of switching matrices [3], exploiting a modified version of the Birkhoff-von-Neumann (BvN) decomposition in which the number of required switching configuration is (almost) minimized, to reduce the negative effects of the reconfiguration time. The original BvN decomposition requires $O(N^2)$ switching configurations, and the modified version requires only $O(N)$ configurations; note that finding the minimum number of configurations is an NP-hard problem [8].

A. Multi-hop Scheduling

We model the multi-hop scheduling problem in two phases:

- *matching-selection*: at the beginning of the f-th scheduling cycle, matching $M(f)$, a set of input-output pairs, is selected, and the optical switching fabric is configured according to it. After a reconfiguration time of R time slots, $M(f)$ becomes active; this configuration is kept for P time slots, i.e. until the scheduling cycle ends.
- *VOQ-selection*: when $M(f)$ is active, the scheduler chooses the VOQ from which packets are transferred. $V(f,n)$, with $n \in [1,P]$, is the set of queues served during the n-th time slot of the f-th scheduling cycle.

The single-hop scheduling, a particular case of multi-hop scheduling, can be modeled as the scheduling where the VOQ-selection is directly determined by the matching selection, whereas, in multi-hop scheduling, the VOQ-selection is decoupled from the matching-selection, since any packet can be sent to a given connected output port regardless of its final destination.

Several algorithms can be devised for the matching and VOQ selection. The aim of this paper is not to determine the best matching and VOQ selection algorithms; rather, we present some simple algorithms which, despite their simplicity, are already able to show some significant performance improvement of the multi-hop approach with respect to the single-hop approach. We better describe the multi-hop approach studied in the paper in the remainder of this section.

A matching M can be represented as a permutation $\pi = (\pi(1)\ \pi(2)\ \dots\ \pi(N))$, of the set O of output ports: $O = \{1,2,\dots,N\}$; $\pi(i)$ denotes the (output) port to which (input) port i is connected. Note that we define $\pi^2(i) = \pi(\pi(i))$, and, by repeatedly applying k times π we can similarly define $\pi^k(i)$. When a packet is present at input i and the matching is kept fixed to π for a scheduling cycle, the packet can be transferred in one hop to port $\pi(i)$, in two hops to $\pi^2(i)$ and in k hops to $\pi^k(i)$. The *hop distance* of input i from output j, given π, is the minimum integer d such that $\pi^d(i) = j$. The *path* between input i and output j, given π, is the sequence of ports by which input i can be connected to output j. Note that a generic permutation does not provide full connectivity among all ports. For example, if $\pi = (2\ 1\ 4\ 3)$, packets at input 1 can be transferred to output 2 (in one hop) and to output 1 (in two hops), but they cannot be transmitted to outputs 3 and 4.

However, it is well known that any permutation can be decomposed in a product of cycles [12]: input i belongs to a cycle of length k (with $1 \le k \le N$) if the distance from itself is k: i.e., $\pi^k(i) = i$. Thus, we define *loop matching* any permutation which is decomposed in a single cycle of length N, that is $\pi^N(i) = i$, for all i. Intuitively, a loop matching is a matching corresponding to a loop among all N ports; thus, it provides full connectivity among all ports with a maximum hop distance of N. For example, when $N = 3$ all the possible loop matchings are: $\pi_1 = (2\ 3\ 1)$, $\pi_2 = (3\ 1\ 2)$; when $N = 4$, $\pi_1 = (2\ 3\ 4\ 1)$, $\pi_2 = (2\ 4\ 1\ 3)$, $\pi_3 = (3\ 4\ 2\ 1)$, $\pi_4 = (3\ 1\ 4\ 2)$, $\pi_5 = (4\ 3\ 1\ 2)$, $\pi_6 = (4\ 1\ 2\ 3)$.

In this work we consider a very simple matching selection: the scheduler chooses a loop matching uniformly at random. Note that the number of all possible loop matchings is $(N-1)!$, corresponding to all the possible way of labelling the nodes in a loop among N nodes. The random choice has complexity $O(N \log N)$ [11].

The main drawback of this matching selection policy is that the average distance among any two ports is $N/2$; thus, on average, the number of hops traversed by each packet is $N/2$. This means that the load offered to an input can increase by a factor $N/2$, and hence the maximum sustainable throughput can be reduced by a factor $N/2$ with respect to the single-hop case. This reasoning is true for a generic VOQ selection, but we will show in Section IV that clever VOQ selection policies are able to sustain a throughput comparable to the case of the single-hop scheduling, but with much shorter delays.

For the VOQ selection, we considered two schemes:

- *k-hop*: each input selects, according to a round-robin scheme, one among the non-empty queues that correspond to ports whose hop distance, according to the selected matching, is not greater than k. The N-hop policy degenerates to a simple round robin policy among all non-empty queues.
- *min-hop*: each input selects the non-empty queue with minimum distance to its destinations.

The multi-hop approach exhibits some drawbacks. When $M(f)$ changes from one scheduling cycle to another, paths among all ports change. Two problems may arise: (i) packets experiencing mis-sequencing, (ii) packets experiencing "vain hops".

Packet mis-sequencing regards packets of the same flow that experience different paths in different scheduling cycles to their destination; hence, packets of the same flow can reach their destination non in FIFO order.

Packets experiencing vain hops are those which have already traveled through many hops across the switching fabric without reaching their final destination; when the matching changes, the new matching may force those packets to travel over a different path, thus requiring an overall too large number of intermediate hops. More formally, define $d_M(i,j)$ as the distance between port i and j when matching M is selected; say that a packet from i to j arrives at port p when M_1 is selected and that later the selected matching becomes M_2, before the packet leaves port p. It may happen that $d_{M_1}(i,p) + d_{M_2}(p,j) > d_{M_2}(i,j)$, that is if the packet was not transmitted during the scheduling period in which M_1 was selected (leaving the bandwidth for other packets), it could have been transmitted later when M_2 was selected, thus, experiencing a shorter distance, and hence increasing the overall throughput.

Note that both problems are more significant for packets following longer paths from the source to the destination; hence both of them can be reduced by giving higher priorities to those packets. One possible solution is to choose, among all the possible packets of a VOQ, the one that has already experienced the largest number of hops, with all packets with the same number of hops served in FIFO order. Thus, we adopt a *per-hop priority queueing* at each VOQ. Packets are grouped into $N+1$ classes, the c-th class corresponding to packets that have been transferred through the switching fabric c times, i.e., that have experienced c hops; the $(N+1)$-th class is for all the packets that have experienced at least N hops. When a VOQ is selected according to the VOQ selection scheme, the packet belonging to the highest priority class is served first.

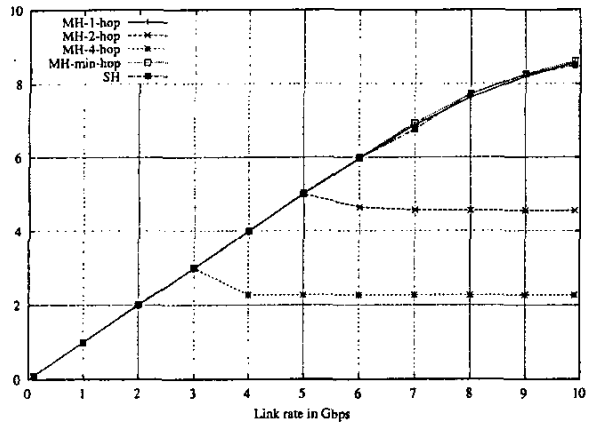

Fig. 1. Output throughput in Gbps under uniform traffic

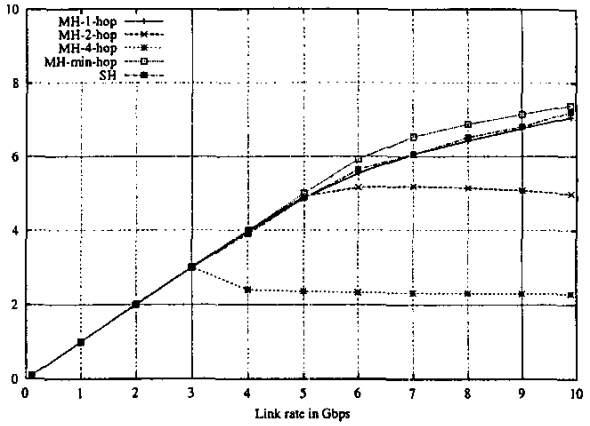

Fig. 3. Output throughput in Gbps under lindiagonal traffic

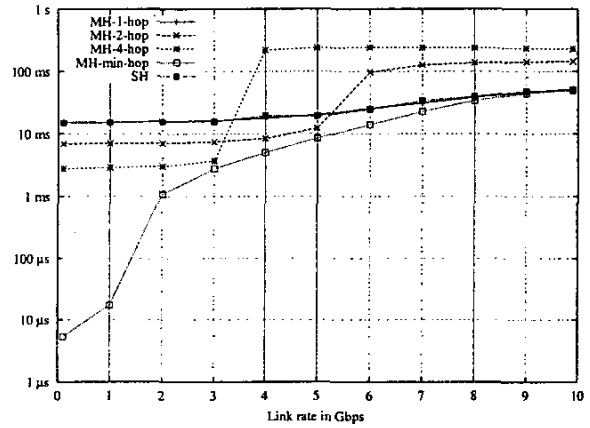

Fig. 2. Average delay under uniform traffic

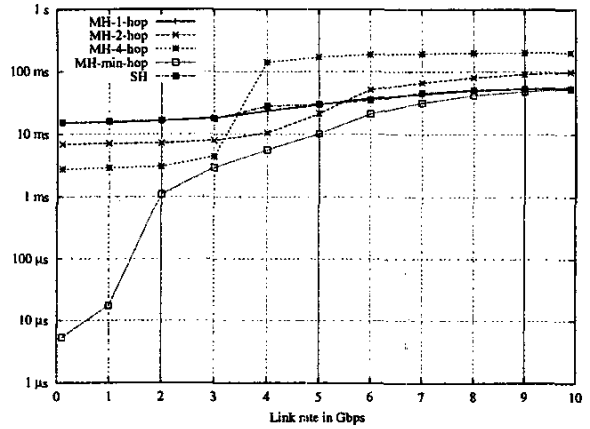

Fig. 4. Average delay under lindiagonal traffic

IV. Simulation results

We compare by simulation, taking as performance indices throughput and delay, multi-hop scheduling algorithms with a single-hop scheduling algorithm. For all Multi-Hop (MH) algorithms, the matching selection is a random choice among loop matchings only, the difference among algorithms being in the VOQ selection scheme; for the Single-Hop (SH) scheduling algorithm the matching selection is a random choice among all possibile matchings. The random choice is by far not optimal for both SH and MH schedulers, but we aim more at a fair comparison among schedulers rather than at performance optimization. We denote by *MH-k-hop* the multi hop scheduler that adopts a k-hop VOQ selection, whereas *MH-min-hop* identifies a min-hop VOQ selection; *SH* is the single-hop (SH) scheduler.

We consider a switch with $N = 16$ ports, with buffer size set equal to 10000 cells for each VOQ, corresponding to about 10 MBytes of fast memory in each line card. We assume a tuning time $R = 0.1$ ms and a link speed $r = 10$ Gbps. For a packet size of 64 bytes, the time slot T lasts 51 ns; thus, the tuning time corresponds to about 2000 time slots. To guarantee a throughput of about 9 Gbps under uniform traffic, the persistence time was set equal to 20000 time slots, i.e. about 1 ms. All numerical results were obtained by stopping simulation runs when a 5% confidence interval width was reached with 95% confidence level.

For the sake of conciseness, we present simulation results for two traffic scenarios only, named uniform and lindiagonal, the former being a classical choice, the latter being a traffic well known to be difficult to scheduled. Formally, denote by ρ the normalized input load, and by $|\cdot|_N$ the modulo N operator; the uniform traffic scenario is such that $\lambda_{ij} = \rho/N$, whereas, for lindiagonal traffic:

$$\lambda_{(i-1),|i+d-2|_N} = \frac{2\rho d}{N(N+1)}$$

where d is the diagonal index of the traffic matrix ($d = 1, \ldots, N$), i.e. the load on each diagonal increases with the diagonal index.

Figs. 1 and 2 show the throughput and the average delays for different scheduling algorithms, as a function of input traffic.

Fig. 1 shows that the k-hop selection limits the throughput to about $1/k$ of the maximum achievable maximum throughput (roughly 9 Gbps); this is the effect of the offered load increase due to packet re-circulations. Thus, only MH-1-hop achieves a throughput close to the the single-hop case. When observing the average delays of k-hop algorithms in Fig. 2, performance become worse as k decrease, showing the tradeoff achievable between throughput and delays. The delays of MH-1-hop are very close to SH, since MH-1-hop is equivalent to a single-hop scheme but in which only loop matchings can be chosen. Since the traffic is uniform, choosing among loop matchings only becomes equivalent to choosing among all possible matchings.

The very high delay experienced by SH can be simply estimated for low load. A lower bound can be computed by considering the fact that, for low load, on average a packet has to wait $N/2 \times (P + R)$, which is 8×1.1 ms=8.8 ms.

MH-min-hop shows the best compromise between throughput and delays, because it achieves the same throughput of SH and, for low loads, delays more than 3 orders of magnitude lower than SH; for medium-high loads, delays are comparable to SH. In a network designed to work with low average utilization (as in the actual Internet core), the performance gain of MH-min-hop with respect to SH is very relevant.

Figs. 3 and 4 show throughput and delays achieved under lindiagonal traffic, to study the adaptability of the schedulers to non uniform traffic.

MH-min-hop outperforms all the other algorithms, showing the best performance both in terms of throughput and delays. Note that all algorithms suffer the fact that the matching (in the single-hop case) or the loop matching (in the multi-hop case) are chosen uniformly at random, and are not adapted to the traffic matrix. Thus, more complex multi-hop (and single-hop) scheduling algorithms, that better tailor the matching selection to the current traffic matrix, should provide better performance that those shown in this paper.

V. Conclusions and Future Work

We presented in this paper a multi-hop scheduling approach that may be beneficial in high-performance large-scale switches based on optical technologies. Multi-hop scheduling implies that packets may traverse several times the switching fabric before leaving the switch. Although this approach may be counter-intuitive, we showed that, when taking into account the potentially large reconfiguration overheads of optical devices, even simple multi-hop scheduling algorithms can significantly outperform traditional scheduling, especially in terms of delays at low and medium loads. Among the presented algorithms, we have showed that the best performance are obtained when controlling and keeping low the number of recirculations for each packet. Out-of-order packet delivery may occur, and although the proposed algorithms are able to keep this phenomenon under control by a proper VOQ selection scheme, we believe that an important future work is finding solutions to completely avoid this phenomenon. Moreover, more clever multi-hop scheduling algorithms could be devised to further improve throughput and delay performance.

References

[1] M. Ajmone Marsan, A. Bianco, E. Leonardi, F. Neri, A. Nucci, "multi-hop Packet Scheduling in WDM/TDM Networks with Nonnegligible Transceiver Tuning Times", *IEEE Trans. on Communications*, Vol.48, n.4, pp.692-703, Apr.2000

[2] T. Anderson, S. Owicki, J. Saxe, C. Thacker, "High speed switch scheduling for local area networks", *ACM Trans. on Computer Systems*, vol. 11, n. 4, Nov. 1993, pp. 319-352

[3] C.S. Chang, W.J. Chen, H.Yi Huang, "Birkhoff-von Neumann Input Buffered Crossbar Switches", *IEEE INFOCOM 2000*, Tel Aviv, Israel, Apr.2000, pp.1614-1623.

[4] S.L. Danielsen, C. Joergensen, B. Mikkelsen, K.E.Stubkjaer, "Optical packet switched network layer without optical buffers", *IEEE Photonics Technology Letters*, vol. 10, n. 6, Jun. 1998, pp. 896-898

[5] P.D. Dobbelaere, K. Falta, S. Gloeckner, "Advances in integrated 2D MEMS-based solutions for optical network applications", *IEEE Optical Communications*, May 2003, pp. 16-23

[6] S. Hengstler, J.J. Uebbing, P. McGuire, "Laser-activated optical bubble switch element" *2003 IEEE/LEOS International Conference on Optical MEMS*, Aug. 2003, pp. 117-118

[7] K. Kar, D. Stiliadis, L.T. Lakshman, T.V., L. Tassiulas, "Scheduling algorithms for optical packet fabrics" *IEEE JSAC*, vol. 21, n. 7, Sept. 2003, pp.1143-1155

[8] I. Keslassy, M. Kodialam, L.T. Lakshman, D. Stiliadis, "On guaranteed smooth scheduling for input-queued switches", *INFOCOM 2003*, vol. 2, Apr. 2003, pp. 1384-1394

[9] I. Keslassy, S.T. Chuang, K. Yu, D. Miller, M. Horowitz, O. Solgaard, N. McKeown, "Scaling internet routers using optics", *ACM SIGCOMM*, 2003, Aug. 2003, pp. 189-200.

[10] X. Li, M. Hamdi, "On scheduling optical packet switches with reconfiguration delay", *IEEE JSAC*, vol. 21, n. 7, Sept. 2003, pp.1156-1164

[11] R. Motwani, P. Raghavan, "Randomized algorithms", *Cambridge Univ. Press*, 1995

[12] H.Schneider, G.P. Barker, "Matrices and linear algebra", Dover Ed., II ed., 1989

[13] B. Towles, W.J. Dally, "Guaranteed scheduling for switches with configuration overhead", *IEEE/ACM Trans. on Networking*, vol. 11, n. 5, pp. 835-847, Oct. 2003