AWG-based optical switches performance using crosstalk limiting schedulers

AWG-based Optical Switches Performance using Crosstalk Limiting Schedulers

D. Fernández Hermida[a], M. Rodelgo Lacruz[b], A. Bianco[c], D. Cuda[c], G. Gavilanes Castillo[d], C. López Bravo[a], F.J. González Castaño[a,b]

[a] *Universidade de Vigo, Spain, davidfh@gti.uvigo.es javier,clbravo@det.uvigo.es*
[b] *GRADIANT, Spain, mrodelgo@gradiant.org*
[c] *Politecnico di Torino, Italy, andrea.bianco,davide.cuda@polito.it*
[d] *Istituto Superiore Mario Boella, Italy, gavilanes@ismb.it*

Abstract

According to a generalized opinion of the scientific community, Arrayed Waveguide Gratings (AWG) are one of the most promising passive optical devices for terabit optical switching systems. These devices permit to build high-performance all-optical WDM switches thanks to their wavelength routing capabilities, in addition to their high information density and lower power consumption. In previous works, we showed that wavelength reuse across different ports introduces in-band crosstalk which strongly limits scalability of AWG-based backplanes. We also proved that this limitation can be overcome by modified scheduling algorithms that reduce the probability of reusing the same wavelength in different ports of the AWG device, significantly reducing or even avoiding the effect of in-band crosstalk. In this paper, we extend several previously proposed scheduling algorithms to enhance their performance. The new algorithms permits to build AWG-based switches of larger sizes while maintaining small bit error rates (BER).

Keywords: Optical Switches, AWG, Scheduling

1. Introduction

The inexorable growth of traffic demand aggravates the bottleneck posed by electronic technology limitations in core network switches. Thus, new optical switching paradigms as Optical Wavelength Switching (OWS) [1], Optical Cell Switching (OCS) [2], Optical Burst Switching (OBS) [3] or Optical Packet Switching (OPS) [4, 5] acquire relevance in the scientific effort.

In the long term, OPS is expected to become the ultimate solution for Wavelength Division Multiplexing (WDM) core networks, thanks to its natural advantages for network control and link bandwidth distribution. A promising line to build high-performance all-optical switches fulfilling OPS requirements is the use of AWG matrices as optical switching fabrics.

As mentioned in [6], AWGs have been very successful in the commercial deployment of WDM transmission systems and they have been recommended for multi-terabit switching devices given that: *i*) they rely on the wavelength dimension to perform switching through the different ports, meaning that they exclusively rely on this optical parameter to produce an actual port change, *ii*) AWGs are relatively mature devices, and thus stable, reliable and commercially available, *iii*) AWGs are relatively simple, behaving as passive multi-port interferometers whose insertion losses depend weakly on the port count, *iv*) AWGs are passive devices, so they do not introduce optical noise and remain transparent with respect to the optical signal. Unfortunately, their scalability strongly depend on coherent crosstalk that limits their maximum size to about 15 input/output ports in the worst-case scenario [7]. The worst-case occurs when the same wavelength is used at the same time in all AWG inputs.

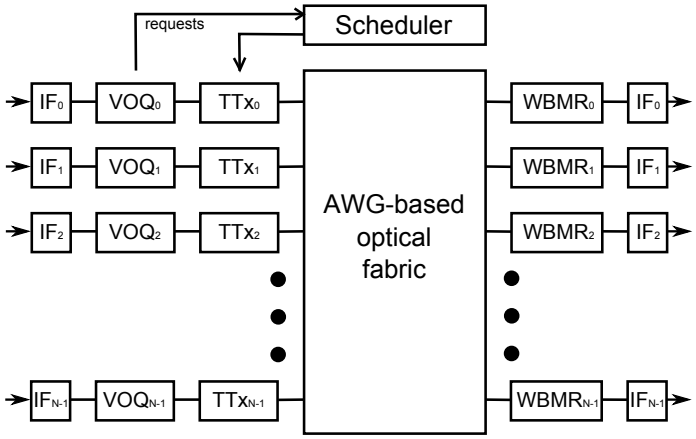


Figure 1: Switch Architecture

Fig. 1 shows the Virtual Output Queuing (VOQ) architecture we are considering, consisting in an AWG connecting N line cards, where each line card is equipped with a fast tunable transmitter (TTx) and a fixed receiver. Fixed-size cells are stored at each input queue in N separate FIFO queues, the VOQs, according to their destination port. At each time slot, a scheduler controlling the input queues selects at most N transfers from the N inputs to the N outputs. As previously mentioned, AWGs passively route optical signals according to their input port and wavelength. Specifically, a packet from input port i using wavelength λ_k is routed to output port j, $j = (i + k) \mod N$.

The occurrence of input-output permutations (input-output port connection patterns) simultaneously employing the same wavelength at different input ports can be avoided with proper packet scheduling. Thus, our objective is to define packet scheduling algorithms with low hardware cost that minimize wavelength reuse (wavelength overlaps) for the signals that cross the AWG at a particular timeslot. This takes us away from the worst-case scenario and permits to obtain

a more realistic upper bound for the maximum size of AWG-based switches. The problem of constraining wavelength reuse in AWG-based switching fabrics has been addressed and solved in [6, 8, 9]: Uniform traffic patterns are admissible in switches without speedup with an odd number of ports, and in switches with an even number of ports with $1 + 1/N$ speedup, where N is the switch size. Nevertheless, in both cases, the solutions are highly complex, thus requiring powerful dedicated hardware. In [6], the authors proposed the use of well-known maximal size matching scheduling algorithms, modified according to the wavelength constraint, with good throughput performance and low hardware cost. This represents a practical alternative for scheduling packets through optical backplanes based on AWGs.

In this paper we explore several modifications of scheduling algorithms to cope with the wavelength reuse restriction and analyze their performance in depth. Throughout the analysis, we consider different wavelength constraints, from the most restrictive, where a single wavelength can be used only once at different input ports during the same timeslot, to less restrictive ones, where a single wavelength can be used at most k times in the same timeslot permutation. This is called a k-legal permutation. In the same way we analyze the probability of wavelength reuse using these modified scheduling algorithms to obtain a more realistic BER and consequently a more precise upper bound for the maximum size of an AWG-based optical switch. The paper is structured as follows. Sec. 2 describes the original maximal size matching scheduling algorithms. Sec. 3 presents the modified packet scheduling algorithms, able to cope with the restriction of wavelength reuse. Sec. 4 validates algorithm performance by means of simulation. Finally, Sec. 5 concludes the paper.

2. 2DRR, iSLIP, and RDSRR

As a basis for the packet scheduling algorithms proposed in this paper, we consider the well-known two-dimensional round-robin (2DRR) [10], iSLIP [11] and Rotating Double Static Round-Robin (RDSRR) [12] schedulers. They show good performance and can be easily modified to handle the wavelength constraint.

2.1. 2DRR

In each timeslot, this algorithm performs N iterations going through the N main diagonals of the traffic matrix and selecting the VOQs with packets to transmit, taking into account that, only one packet can be extracted from a particular input port and only one packet can be sent to a particular output port in a given timeslot. For a fair VOQ selection, at each timeslot, the first diagonal to be checked is modified. Thus, each input/output pair has the maximum priority once every N timeslots, avoiding possible starvation issues. For instance, Fig. 2 shows the diagonal searching order at timeslot t.

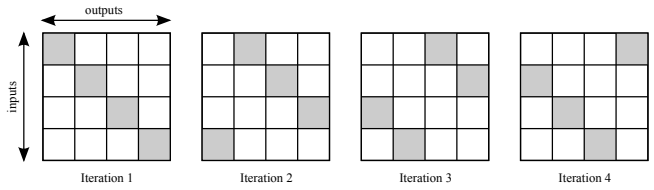


Figure 2: 2DRR iterations in timeslot t

2.2. iSLIP

iSLIP is considered a *de facto* standard for iterative maximal size matching algorithms [11, 13]. Matchings are computed through an iterative procedure. Each iteration can be divided in three phases, as shown in Fig. 3: *i*) *Request*: each unmatched input request the outputs for which it has a queued cell; *ii*) *Grant*: every output module selects one of the requests and notifies the inputs whether or not their requests were granted; *iii*) *Accept*: every input module selects one of the received grants and notifies the outputs whether or not their grants were accepted. To avoid contention, it is necessary to establish priority rules through arbiters (pointers). Every output/input module has a pointer that selects the input/output with higher priority in a particular timeslot. For the sake of fairness, the pointer arbiters must change value while time progress. The pointer updating rule of iSLIP states that pointers are updated to the next position after the accepted input/output.

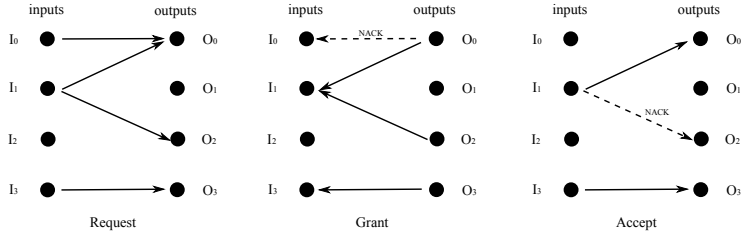


Figure 3: iSLIP phases

It has been shown that, in practice, iSLIP converges in $O(\log_2 N)$ iterations [13], where N is the switch size.

2.3. RDSRR

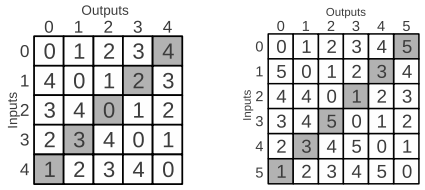
The Rotating Double Static Round-Robin (RDSRR) algorithm is very similar to iSLIP but differs from it in the pointer updating rule. In RDSRR, the pointers are incremented (modulo N) after each timeslot, regardless of the packet assignments, to maintain the priority pattern with different wavelengths. Furthermore, the search direction is reversed at each timeslot to improve fairness in case of non-uniform traffic.

3. λ-2DRR, λ-iSLIP, and λ-RDSRR

The previous algorithms were designed to maximize the throughput in a VOQ system with the sole restriction of no speedup. Only one packet can be taken from an input port and only one packet can be sent to an output port at each timeslot. This condition alone does not imply that the wavelength reuse constraint needed to deal with the issue of coherent crosstalk in AWG-based switches is satisfied. Thus, to avoid excessive levels of crosstalk, it is also necessary to restrict the number of times a single wavelength is reused in each timeslot. Thus, we modify the previous algorithms to introduce a parameter that restricts the set of legal permutations (input-output connection patterns) on the basis of the constraint on the number of wavelength reused.

3.1. λ-2DRR

λ-2DRR is an adaptation of the two-dimensional round-robin (2DRR) scheduler [10] that satisfies the wavelength reuse restriction, taking into account that, as shown in [8], the switching patterns that minimize wavelength reuse occur when the queues corresponding to anti-diagonals elements are selected. Specifically, at each timeslot, λ-2DRR scans the request matrix sweeping through all the N anti-diagonals, looking for queued cells in the VOQs. The algorithm starts searching in the anti-diagonal pointed by the arbiter and selects one packet from $VOQ_{i,j}$ in that anti-diagonal if: i) there is at least one packet in that queue; ii) input i and output j are both available; and iii) the corresponding wavelength has not been selected more than k times in previous iterations on this timeslot. In the next step, the algorithm selects the next anti-diagonal and repeats the search process. The algorithm ends when all N anti-diagonals were analyzed.



(a) 1-legal on odd N traffic matrix

(b) 2-legal on even N traffic matrix

Figure 4: Main antidiagonal wavelength reuse property (the numbers indicate the corresponding wavelength index k)

To ensure fairness, at each timeslot λ-2DRR changes the first anti-diagonal to be checked. **Fig. 4 shows the AWG tuning matrix. The numbers in the boxes indicate the wavelength index $k = (j - i) \bmod N$ that is required to transmit the packet from input port i to output port j. If the switch size is odd, all the connections of the same anti-diagonal (shaded boxes) use different wavelength indexes and all the connections can be selected in parallel. However, if the switch size is even,**

as demonstrated in [8], it is not possible to find a connection pattern in which all the wavelengths are different. Thus, each wavelength appears twice in every anti-diagonal. If each wavelength can be used only once, the connection must be selected according to some rule. To this goal we employ an output arbiter that points to the output with highest priority in that particular timeslot. Both arbiters (the one to select the anti-diagonal and the one to give priority to outputs) follow a round-robin strategy.

3.2. λ-iSLIP selecting lambdas at outputs

In this extension to iSLIP labeled as λ-iSLIP@out, to ensure the wavelength reuse constraint each of the N available wavelengths is associated with a λ-pointer that manages requests and grants for its associated wavelength. Each output is equipped with a p_j-pointer that manages priorities among inputs. In the same way, each input is equipped with a p_i-pointer that manages priorities among outputs. At each timeslot, pointers are initialized to give priority to input-output pairs corresponding to one of the anti-diagonals of the traffic matrix for N odd. When N is even, if we select the anti-diagonals all lambdas present in the anti-diagonal would be repeated twice (see Fig. 4). Thus, we look for patterns that minimize wavelength reuse. As shown in Fig. 5 we can divide the λ-matrix into three parts: *i*) columns from 0 to $N/2$, *ii*) columns from $N/2+1$ to $N-2$, and *iii*) the last column. **The pointers are initialized as shown in Eq.**(1) **(for output pointers), and Eq.**(2) **(for input pointers). Wavelength pointers are initialized to the VOQs with highest priority. Thus, only two cells in the maximum priority pattern share a wavelength. This pair of VOQs has less transmission opportunities (i.e. if $k = 1$ only one of them will be able to transmit). Once initialized, the maximum priority pattern is shifted every time slot to give all VOQs the same transmission opportunities. Furthermore, every time slot, most priority cells are shifted up one position to rotate the underprivileged pair (same wavelength) among the inputs, and every N timeslots the cells with highest priority are shifted one position to the right to rotate that pair among the outputs. Thus, every N^2 slots, the highest priority patterns is repeated, and all the input-output pairs in the maximum priority pattern share their wavelength twice with other VOQ in the pattern.**

$$p_j = \begin{cases} (-1-j) \mod N & \text{if } j \in [0, \frac{N}{2}-1] \\ (-2-j) \mod N & \text{if } j \in [\frac{N}{2}, N-2] \\ \frac{N}{2}-1 & \text{if } j = N-1 \end{cases} \quad (1)$$

$$p_i = \begin{cases} (-2-i) \mod N & \text{if } i \in [0, \frac{N}{2}-2] \\ N-1 & \text{if } i = \frac{N}{2}-1 \\ (-1-i) \mod N & \text{if } i \in [\frac{N}{2}, N-1] \end{cases} \quad (2)$$

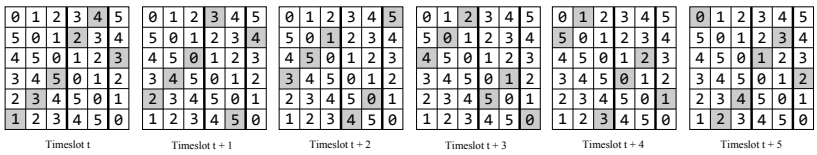


Figure 5: Evolution of pointer arbiters

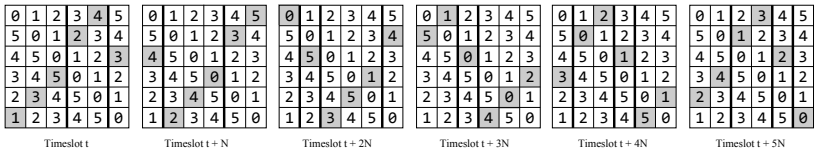


Figure 6: Evolution of pointer arbiters each N timeslots

For odd N, input pointers are initialized to $p_i = (-1 - i) \mod N$, output pointers are initialized to $p_j = (-1 - j) \mod N$, and lambda pointers are initialized to the corresponding value to give priority to the outputs that are pointing to them. The updating rule followed by this algorithm is to change the anti-diagonal arbiters are pointing to. Thus, input and output pointers are incremented by one (modulo N), and the lambda pointers are updated so that they point to the output pointing to them.

3.3. λ-iSLIP selecting lambdas at inputs

The algorithm labeled as λ-iSLIP@in is very similar to λ-iSLIP@out, but in this case, instead of using the input/output matrix to select the anti-diagonal, we use the output/lambda matrix. Only a pointer permutation is needed. Thus, we initialize output pointers to $p_j = (-1 - j) \mod N$, lambda pointers to $p_k = (-1 - k) \mod N$, and input pointers point to the proper outputs to give priority to the selected anti-diagonal.

3.4. λ-RDSRR selecting lambdas at outputs

This algorithm (λ-RDSRR@out) differs from λ-iSLIP@out in two main aspects, the pointer updating rule and signaling order. Inputs have p_i-pointers that are initialized to $p_i = (-i) \mod N$, outputs p_j-pointers are initialized to $p_j = (-j) \mod N$, and lambdas p_k-pointers (λ-pointers) initialized to $p_k = (-k) \mod N$. **Therefore, if an input pointer p_i points to an output port j, output port pointer p_j reciprocally points to input port i. Since the packet grants and packet accepts start from pointer positions, if there is a packet in a pointed input port destined to a pointed output port, it will be selected by the scheduling algorithm independently of the switch state. So, that input-output connection has maximum priority. Moreover, if the next output port of pointer p_i is output port**

j' the next port of pointer $p_{j'}$ is input port i. Thus, those connections have second order priority and so on. The wavelength pointers are initialized in such a way that they point to a maximum priority VOQ, the next VOQ from the pointer has the second priority level, and so on. Initially all input modules send signals to those outputs for which they have queued cells. Then, each output selects one request (if any) and grants it to the corresponding input. Each input selects one granted request received and forwards one request to the corresponding lambda arbiter that manages requests and grants for its wavelength. The lambda arbiter grants the requests depending on the k-legal constraint and following the pointer order. If the input receives a grant from the lambda arbiter, then it sends an accepts to the output. At the end of each timeslot the pointers are updated in the following way: input and output pointers are incremented by one (modulo N), λ-pointers are decremented by one (modulo N). Note that the only difference between wavelength and output arbiters is that the former grant k requests and the latter just one. Since they do not work at the same time, they can be easily implemented as the same physical module with minimal extra cost.

3.5. λ-RDSRR selecting lambdas at inputs

λ-RDSRR@in differs from λ-RDSRR@out in the order of the steps. First all inputs send requests to the corresponding lambdas of the outputs for which they have queued cells. Then, each lambda selects requests according to the k-legal constraint and responds to the outputs. A given output selects one of the received requests and grants it to the associated input. The inputs select and accept grants from the outputs and the corresponding lambdas. The pointer initialization and updating rule is the same as λ-RDSRR@out.

4. Results

In this section we present performance results obtained by simulation. Each VOQ has a capacity of 1 million cells each; at simulation startup, all queues are empty. We analyze the evolution of the queues at each timeslot on a time horizon of 10 million slots. The first 100000 slots correspond to transient time, and data collected within this time do not count for the calculation of the statistics. As a reference, we compare the previously presented algorithms with the standard iSLIP, RDSRR and 2DRR algorithms, which do not take into account the wavelength reuse constraint.

4.1. Packet delay evaluation

We study the algorithm behavior in terms of delay considering two types of traffic: uniform and log-anti-diagonal traffic.

4.1.1. Uniform Traffic

Under the uniform traffic assumption, each input receives independent and identically distributed traffic with the same load.

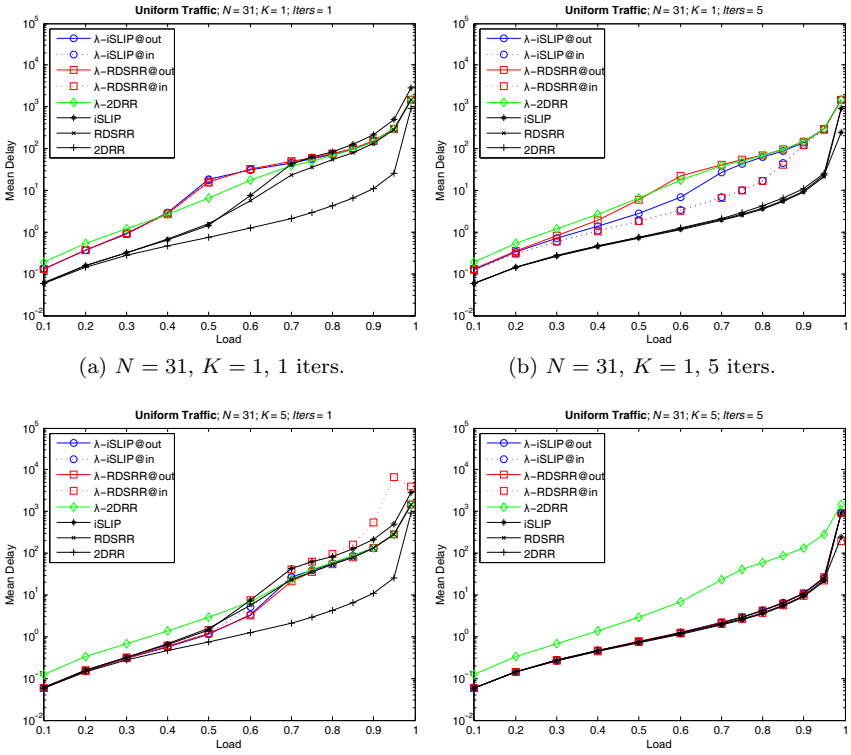
4.1.1.1 AWG matrix with odd number of ports

Fig.7 shows the simulation results in four different scenarios, for $N = 31$. In Figs. 7a and 7b no wavelength reuse is allowed, while in Figs.7c and 7d the same wavelength can be reused at different port up to five times, in the same timeslot. Figs.7a and 7c show results for a single iteration, and Figs.7b and 7d for five iterations (in the order of $\log_2 N$ iteration). Simulations of the λ-2DRR algorithm always perform $N = 31$ iterations due to the algorithm definition. For a single iteration, all algorithms perform very similarly at high traffic loads, above 70%. However, when the number of iterations increases, the algorithms that select lambdas at the input introduce lower delay. Fig.7d shows that all algorithms but λ-2DRR perform very similarly with a maximum wavelength reuse of five and using five iterations. $\lambda-$2DRR behavior is less sensitive to the number of iterations allowed, and only a slight improvement is introduced when the allowed wavelength reuse is five. Results in Fig.7d are very close to those of the simulations allowing full wavelength reuse and 31 iterations and thus, this result is a considerable reduction of the physical impairment (from full wavelength to 5 reuses), under equivalent scheduled packet delay. Indeed, when few iterations are allowed, delays of proposed λ-algorithms are close to the original schedulers which do not consider the wavelength constraint.

4.1.1.2 AWG matrix with an even number of ports

When the number of inputs/outputs of a switch is even, there are not 1-legal connection patterns that permit achieve 100% throughput, as shown in [8]. This limits the maximum throughput achievable if we limit the number of wavelength reuse.

Figs.8a, 8b, and 8c show that λ-RDSRR@in begins dropping packets at low loads for a single iteration. The same occurs with λ-RDSRR@out with a single wavelength reuse. Thus, both algorithms perform badly for a single iteration and a single wavelength reuse and incrementing the number of iterations improves λ-RDSRR@in performance, whereas incrementing the number of allowed wavelength reused improves λ-RDSRR@out performance. As in the odd number of port case, when a small number of iterations and a low wavelength reuse are allowed, performance is close to the one ensured by the original version of the scheduling algorithm which do not cope with the wavelength constraint.



(a) $N = 31$, $K = 1$, 1 iters.

(b) $N = 31$, $K = 1$, 5 iters.

(c) $N = 31$, $K = 5$, 1 iters.

(d) $N = 31$, $K = 5$, 5 iters.

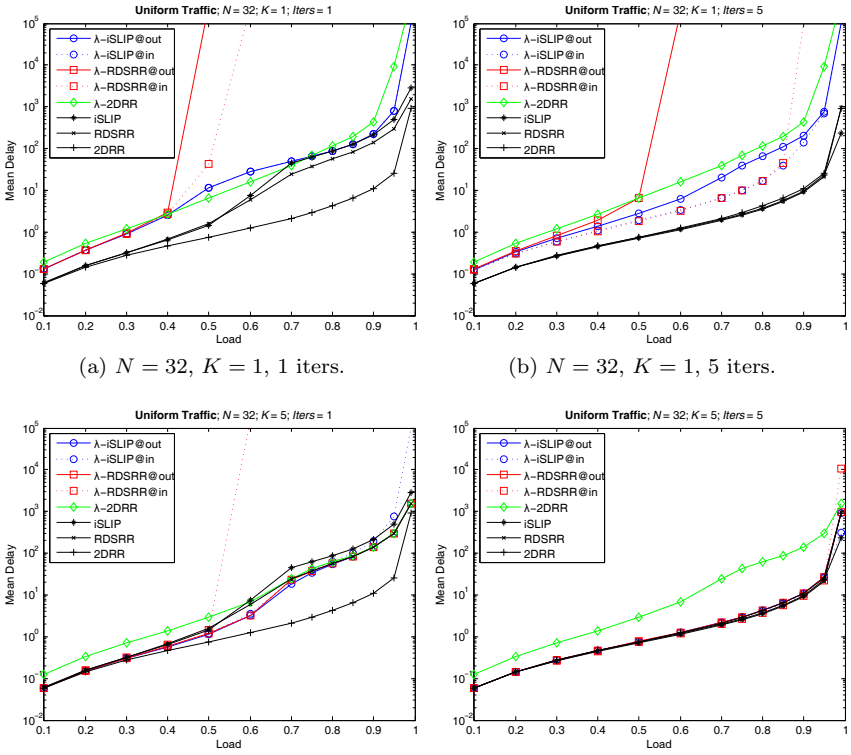
Figure 7: Uniform traffic, $N = 31$

4.1.2. Unbalanced traffic

The log-antidiagonal traffic schema offers different loads to each input/output pair and is described by the following matrix, for $N = 5$:

2^3	2^2	2^1	2^0	2^4
2^2	2^1	2^0	2^4	2^3
2^1	2^0	2^4	2^3	2^2
2^0	2^4	2^3	2^2	2^1
2^4	2^3	2^2	2^1	2^0

This matrix shows the proportion of the load assigned to each input/output connection. The reason for choosing this type of traffic is to test the contrast produced by the rotary behavior through the diagonals of the algorithms (i.e. 2DRR) in anti-diagonal traffic scenario. This represents a more realistic traffic than a simple diagonal traffic test case (due to its logarithmic relationship), and being more severe than uniform traffic (due to the diagonal unbalance). The results for this input traffic are shown in Figs. 9 and 10 for switch sizes $N = 31$



(a) $N = 32$, $K = 1$, 1 iters.

(b) $N = 32$, $K = 1$, 5 iters.

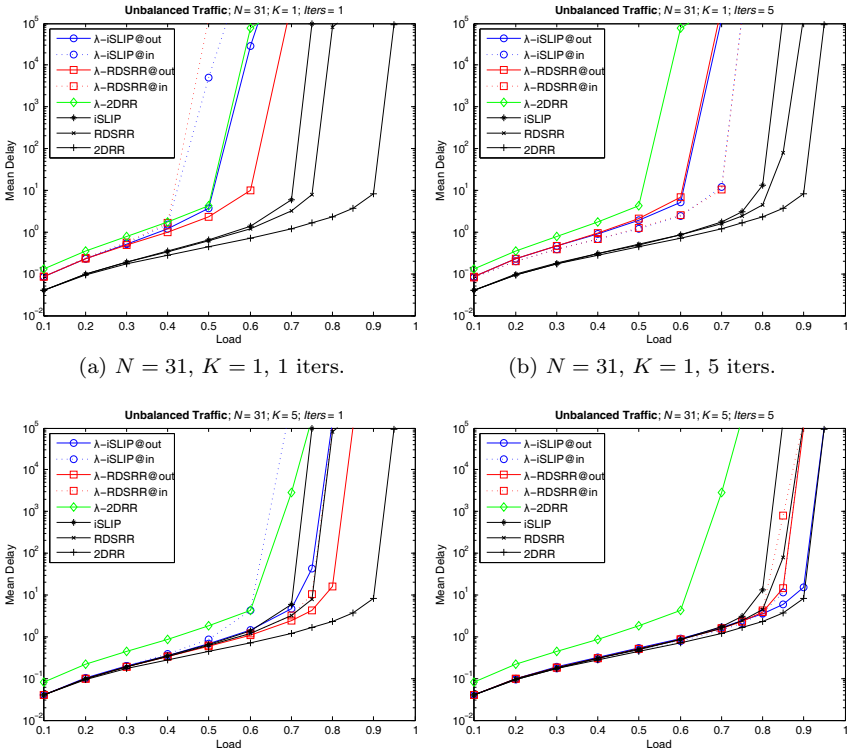
(c) $N = 32$, $K = 5$, 1 iters.

(d) $N = 32$, $K = 5$, 5 iters.

Figure 8: Uniform traffic, $N = 32$

and $N = 32$, respectively. The large difference between the probabilities of having a non-empty queue on the main anti-diagonal with respect to the remaining VOQs, expose packets outside the main anti-diagonal to longer delays and higher losses, creating unfairness issues. As shown in Fig.9d, the algorithms based on iSLIP and RDSRR perform better than λ-2DRR. Also, the proposed algorithms perform better than iSLIP itself. This is because the pointer updating rule reinforces selecting packets in the anti-diagonals. The log-antidiagonal traffic offers packets across the main anti-diagonal with high probability.

Fig.9a shows that when we only allow 1-legal permutations and a single iteration, the algorithms that select lambdas at the inputs perform worse than those that select lambdas at the outputs. Nevertheless, if we increase the number of iterations this situation is reversed, and the algorithms that select lambdas at the inputs reach better results. The results in Fig.9 and 10 are very similar. Under unbalanced traffic, outputs are more critical than wavelengths (in the extreme case, every packet is destined to just one output and the associated wavelengths are all different). For a single iteration, if wavelengths are selected first, many selections are discarded in the next step because they have the



(a) $N = 31$, $K = 1$, 1 iters. (b) $N = 31$, $K = 1$, 5 iters.

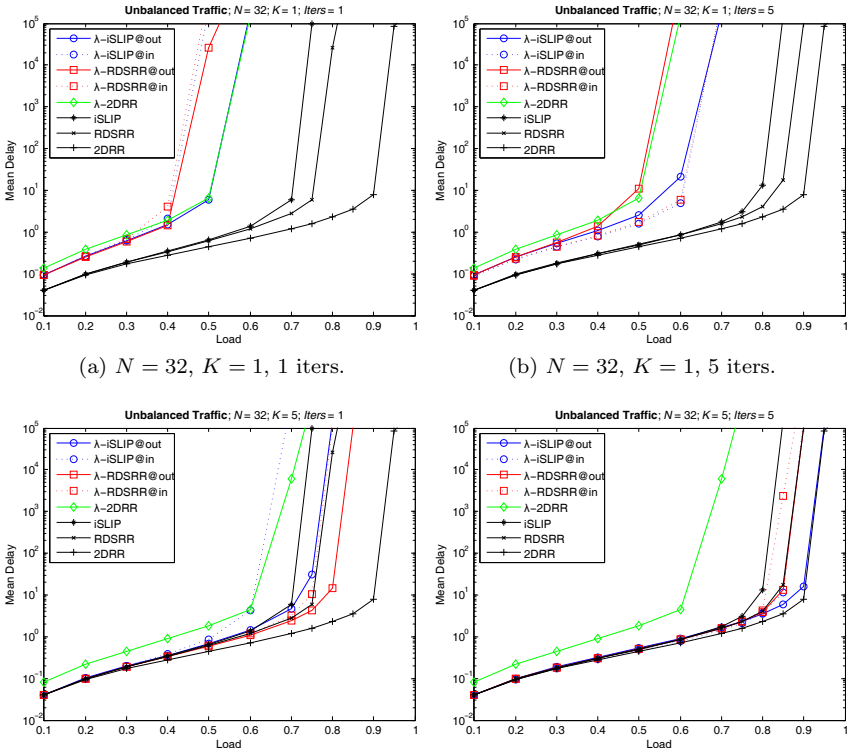
(c) $N = 31$, $K = 5$, 1 iters. (d) $N = 31$, $K = 5$, 5 iters.

Figure 9: Unbalanced traffic, $N = 31$

same output. On the other hand, if we select the outputs first, the probability that such a selection implies different wavelengths is higher. If we increase the number of iterations, the assignments discarded in the second step (wavelength or output) are assigned in subsequent iterations. In this case, as under uniform traffic, it is preferable to select the wavelengths first, because this leads to a more distributed packet assignment among different timeslots. We can conclude then that with a small number of iteration and a a small wavelength reuse, performance is close to performance ensured by classic scheduler which do not take into account the wavelength reuse. Hence, performance are mainly limited by the scheduler itself which is not able to achieve 100 % of throughput in the case of the unbalanced traffic scenario. Thus the wavelength constraint does pose a practical limit on the switch size.

4.2. Wavelength reuse distribution

Intrinsically, the algorithms presented in the previous section try to select switching patterns that minimize wavelength reuse probability, hence reducing coherent crosstalk to permit to scale to larger switch size for a fixed BER. In this



(a) $N = 32$, $K = 1$, 1 iters. (b) $N = 32$, $K = 1$, 5 iters.

(c) $N = 32$, $K = 5$, 1 iters. (d) $N = 32$, $K = 5$, 5 iters.

Figure 10: Unbalanced traffic, $N = 32$

section, we analyze the reuse distribution for uniform and unbalanced traffic, at different loads, for all algorithms and switch sizes of 31 and 32. The results allow us to assess the wavelength reuse in case it is unbounded and to determine more realistic size bounds for AWG-based switches than those obtained in [7].

To analyze the reuse distribution, **we run the simulations without any reuse constraint, i.e., we simulate the algorithms with k-legal** constraint set to the switch size. Thus, wavelengths can be fully reused at any port. At each timeslot, after the scheduling part we check the number of times each wavelength is reused. Thus, for different traffic loads we can plot the distribution of the wavelength reuse.

Figs. 11 and 12 show the results for uniform and unbalanced traffic, respectively. The height of each color bar indicates the probability that a packet is transmitted with a given wavelength overlap (i.e. the dark blue "1-overlap" bar indicates the probability that a packet is transmitted in a wavelength that is not employed by any other input ports). For a switch size of 31 and uniform traffic, all algorithms exhibit a similar distribution with the highest probability of large reuse factor somewhere

around 90%-95% load. This is caused by the behavior of the algorithms at high loads. Since the probability of empty VOQ tends to zero at high loads, the algorithms select anti-diagonal VOQs with higher probability, i.e. those with single (one) overlap pattern connection. The behavior of λ-2DRR is a little bit different. The probability of having one wavelength used begins to increase at medium loads, around 60%. This behavior is produced as a consequence of more delayed packets. This means queues with more packets stored and then the probability of empty queues decreases, thus increasing the probability of selecting packets on the anti-diagonals.

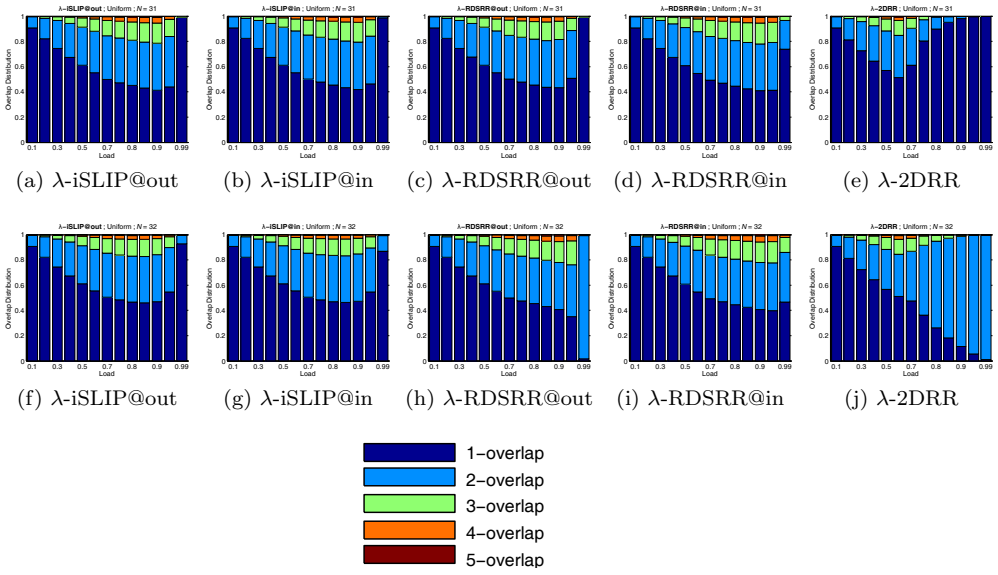
For a switch size of 32 and uniform traffic, the distribution of the wavelength reuse differs slightly for the considered algorithms. In the case of λ-RDSRR with selection of lambdas at the outputs, at high loads the algorithm selects with higher priority the anti-diagonals where the pattern has two wavelengths reused, then the probability of having wavelength reused twice increases. The same holds for λ-2DRR. The distributed iSLIP algorithm avoids this issue by initializing the pointers at each iteration, selecting with higher priority those patterns with just one wavelength reused twice.

The results for unbalanced traffic are very similar. At low and medium loads, the probability of large overlaps slightly decreases because packets concentrate in a few anti-diagonals which are assigned to all-different wavelengths for $N = 31$ and almost all-different for $N = 32$. At high loads, the observed decreasing overlapping probability for uniform traffic does not occur since under unbalanced traffic, the VOQs having less traffic are empty more frequently.

In general, under both traffic patterns the wavelength reuse is far below the worst case and the probability of wavelength reuse above 5 overlaps is lower than 0.2%. Thus, the pointer initialization and evolution of the algorithms can naturally reduce wavelength reuse and the resulting coherent crosstalk would be low enough for most practical implementations. This may permit to avoid the wavelength selection stages, simplifying the algorithm and reducing hardware implementation complexity.

5. Conclusions

In this paper we proposed several packet scheduling algorithm variants with the objective of minimizing the probability of wavelength reuse in switch connection patterns to reduce coherent crosstalk in AWG-based switches. This permits to increment switch size, under the constraint of a given BER. Results show that the proposed algorithms provide good delays, even with severe restrictions on the number of wavelengths that can be reused. Constraining the wavelength reuse to $\log_2 N$ does not worsen scheduling performance with respect to those obtained without any constraint. Thus, considering that an AWG can support up to sixteen wavelengths reused before the BER exceeds 10^{-12}, it is possible to build switches with sizes of up to 216 inputs and outputs. From a practical point of view, this means that there is no real limit when the proposed algorithms are used to schedule arriving packets. We compared the algorithms changing the



(a) λ-iSLIP@out (b) λ-iSLIP@in (c) λ-RDSRR@out (d) λ-RDSRR@in (e) λ-2DRR

(f) λ-iSLIP@out (g) λ-iSLIP@in (h) λ-RDSRR@out (i) λ-RDSRR@in (j) λ-2DRR

(k) Legend

Figure 11: Wavelength reuse distribution for uniform traffic. $N = 31$ above $N = 32$ below.

wavelength selection side. Selecting wavelengths at the inputs perform better for several iterations when the reuse restriction is very severe. Instead, when the wavelength reuse constraint is relaxed and few iterations are performed, wavelength selection at outputs provides enhanced performance.

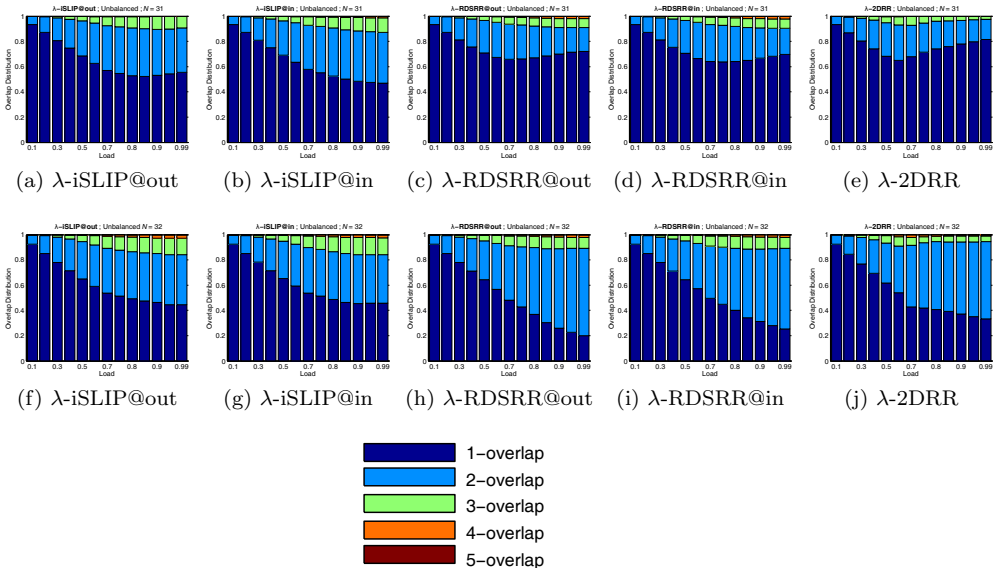
6. Acknowledgments

This work was supported by the PHOBOS 09TIC014CT grant (Xunta de Galicia, Spain), by the CALM TEC2010-21405-C01 grant (Ministerio de Ciencia e Innovación, Spain), and by the FIERRO Network TEC2010-12250-E grant (Ministerio de Ciencia e Innovación, Spain).

This work was started on the basis of an original idea of Fabio Neri. Fabio suddenly passed away in April 2011. We wish to dedicate this paper to his memory.

7. References

[1] I.P. Kaminow, C.R Doerr, C. Dragone, T. Koch, U. Koren, A.A.M Saleh, A.J. Kirby, C.M. Ozveren, B. Schofield, R.E. Thomas, R.A. Bany, D.M. Castagnozzi, V.W.S Chan, B.R. Hemenway, D. Marquis, S.A. Parikh, M.L.



(a) λ-iSLIP@out (b) λ-iSLIP@in (c) λ-RDSRR@out (d) λ-RDSRR@in (e) λ-2DRR

(f) λ-iSLIP@out (g) λ-iSLIP@in (h) λ-RDSRR@out (i) λ-RDSRR@in (j) λ-2DRR

(k) Legend

Figure 12: Wavelength reuse distribution for unbalanced traffic. $N = 31$ above $N = 32$ below.

Stevens, E.A. Swanson, S.G. Finn, R.G. Gallager, A Wideband All-Optical WDM Network. In IEEE Journal on Selected Areas in Communications, vol.14, no.5, pp.780-799, June 1996.

[2] H.J. Chao, S.Y. Liew, A New Optical Cell Switching Paradigm. In Proceedings of International Workshop Optical Burst Switching, WOBS 2003, Dallas, Texas, USA, October 2003.

[3] C. Qiao, M. Yoo, Optical Burst Switching (OBS) - A New Paradigm for an Optical Internet. In Journal of High Speed Networks, vol. 8, pp. 69-84, 1999.

[4] S. Yao, B. Mukherjee, S. Dixit, Advances in Photonic Packet Switching: an Overview. In IEEE Communications Magazine, vol. 38, no. 2, pp. 84-94, 2000.

[5] D.K. Hunter, I. Andonovic, Approaches to Optical Internet Packet Switching. In IEEE Communications Magazine, vol. 38, no. 9, pp. 116-122, 2000.

[6] A. Bianco, D. Cuda, G. Gavilanes-Castillo, F. Neri, M. Rodelgo-Lacruz, F.J. Gonzlez-Castao, C. Lpez-Bravo, M. Salvat, Crosstalk Limiting Schedulers in AWG-based Optical Switches. In IEEE Global Telecommunications Conference, Miami, Florida, USA, December 2010.

[7] J. M. Finochietto, R. Gaudino, G. A. Gavilanes, and F. Neri. Simple optical fabrics for scalable terabit packet switches. In IEEE International Conference on Communications, Beijing, China, May 2008.

[8] M. Rodelgo-Lacruz, C. Lopez-Bravo, F.J. Gonzalez-Castano, H.J. Chao, Practical scalability of Wavelength Routing Switches. In IEEE International Conference on Communications, Dresden, Germany, June 2009.

[9] A. Bianco, D. Hay, F. Neri, Crosstalk-Preventing Scheduling in Single- and Two-Stage AWG-Based Cell Switches, in IEEE/ACM Transactions on Networking, vol.19, no.1, pp.142-155, February 2011.

[10] L.O. LaMaire, D.N. Serpanos, Two dimensional round-robin schedulers for packet switches with multiple input queues. In IEEE/ACM Transactions on Networking, vol. 2, no. 5, pp. 471-482, October 1994.

[11] N. McKeown, iSLIP: A scheduling algorithm for input-queued switches. In IEEE/ACM Transactions on Networking, vol. 7, no. 2, pp. 188-201, April 1999.

[12] Y. Jiang, M. Hamdi, A fully desynchronized round-robin matching scheduler for VOQ packet switches with multiple input queues. In IEEE/ACM Transactions on Networking, vol. 2, no. 4, pp. 471-482, October 1994.

[13] T. E. Anderson, S. S. Owicki, J. B. Saxe, C. P. Thacker, High speed switch scheduling for local-area networks. In ACM Transactions on Computer Systems, vol. 11, no. 4, pp. 319-352, November 1993.