Packet reordering in the Lightpath Bundling & Anycast Switching (LB+AS) paradigm

Jose-Luis Izquierdo-Zaragoza, Pablo Pavon-Marino
Department of Information Technologies and Communications
Universidad Politécnica de Cartagena
Cuartel de Antiguones, Plaza del Hospital 1, 30202 Cartagena, Spain
{josel.izquierdo, pablo.pavon}@upct.es

***Abstract*—In optical networks, Lightpath Bundling & Anycast Switching paradigm (LB+AS) consists of (i) bundling together the lightpaths which follow a common route (LB), and (ii) perform a fine-grained per-packet granularity balance of the traffic among the lightpaths in the bundle (AS). Recent works suggest that, with seamless changes in the electronic equipment, and no changes in the optical infrastructure, LB+AS can bring promising advantages to existing optical networks. In particular, AS operation spreads the traffic among the lightpaths in the bundle, to enhance the statistical multiplexing gain. However, this type of operation could bring up packet reordering problems if different packets of the same flow traverse different lightpaths. Packet reordering issues especially affect protocols such as TCP, reducing the goodput and increasing the flow completion time. This paper addresses this problem and investigates viable implementations of anycast switching techniques that: (i) allow a practical implementation in commercial high-speed switches, and (ii) eliminate the packet reordering problem in practice. Among several solutions presented and tested, the e-JSQ scheme has emerged as a solution fulfilling both requirements.**

Keywords—optical networks; lightpath bundling; anycast switching

I. INTRODUCTION

In multilayer WDM networks, higher layer (i.e. IP) traffic is transported onto a virtual topology of all-optical paths called lightpaths. Lightpaths occupy one wavelength in each traversed fiber. In multilayer WDM networks, Lightpath Bundling (LB) is a control plane technique which consists of grouping together the lightpaths between two nodes that follow a common route (in any wavelength), so that electronic layers (e.g. IP) see them as a single entity of aggregated capacity. Anycast Switching (AS) is, in its turn, a data plane technique to be implemented in the electronic switch fabrics of the routers. It consists of instructing the fabrics to perform a fine-grained *per-packet* load balance of the traffic among the lightpaths bundled. This is the source a significant statistical multiplexing gain, and thus a reduction of the packet delay and buffering needs in the routers [1].

A recent work presented in [2] shows that the combined application of Lightpath Bundling and Anycast Switching, would significantly improve the economic efficiency of multilayer optical WDM networks: (i) reducing CAPEX/OPEX while maintaining the revenue (the same carried traffic), or (ii) increasing the revenue with no CAPEX/OPEX reduction. This would require no changes in the optical infrastructure (fibers, line equipment, ROADMs), and seamless changes in the electronic data plane and the control plane.

From the point of view of the electronic switching, a key feature of AS is that traffic is distributed at a *per-packet granularity* among all the output ports corresponding to the lightpaths in the bundle. This is a different approach with respect to classical per-flow traffic balance, which has been proposed for other load balancing contexts like Ethernet Link Aggregation (802.1AX) or IP Equal-Cost Multi-Path technique (RFC 2992 [3]), or more sophisticated approaches such as flowlet-switching [4]. In particular, a per-flow balancing requires identifying the flow of every packet at wire-speed, to switch all the packets in the same flow through the same output port. Splitting the traffic at a per-packet granularity gives a finer balance (and thus better statistical multiplexing gain) and avoids the computational cost of wire-speed identification. However, it has the obvious downside of potential packet reordering, since consecutive packets within a given flow may now traverse different lightpaths (in the same bundle) to the destination.

Packet reordering can have a heavy effect on the performance of some protocols in the Internet. In particular, in a TCP connection the reception of out-of-sequence data packets results in the transmission of duplicate ACKs (DUP-ACKs). When three or more consecutive DUP-ACKs are received by the TCP source, it assumes that a packet loss occurred, reducing the congestion window and thus the rate. In summary, out-of-sequence packets can lead to false over-estimation of the network congestion.

In a previous work [5] we investigated the performance of TCP flows in a LB+AS network. We were interested in complementing the work [2] (which assumed inelastic traffic sources), by assessing whether the fine-grained traffic balancing of LB+AS was also beneficial for TCP elastic traffic sources. Our results in [5] showed that LB+AS was also in this context able to reduce the queuing delay, the buffering needs in the nodes, and as a side-effect, increase the throughput and lower the drop rate of long-lived TCP flows. These results suggest that LB+AS would be TCP-friendly. Intuitively, this is a logical result: balancing the traffic at a packet granularity is

This research was partially supported by the Spanish project grant TEC2010-21405-C02-02/TCM (CALM) and the predoctoral fellowship program of the Universidad Politécnica de Cartagena. It was also developed in the framework of the project "Programa de Ayudas a Grupos de Excelencia de la Región de Murcia" funded by F. Séneca (Plan Regional de Ciencia y Tecnología 2007/2010).

beneficial by nature, since all the flows traversing a bundle are aggregated, spreading out bursts [6].

However, the work in [5] assumed that AS was implemented in the electronic routers using an *ideal* Join Shortest Queue (JSQ) technique, where each packet is switched to the output port with less traffic pending to be carried. As will be shown later, this is not viable in conventional high-speed electronic switches like those based on CIOQ (Combined Input/Output Queuing) schemes ("all" the fabrics in the backbone). The reason is that an ideal JSQ technique requires the input line cards to decide on the output port of the packet using information that is not local: the occupation of the queues at the output line cards.

In this paper, we investigate for the first time several AS techniques that try to alleviate the potential packet reordering issues, and that have a *practical* implementation in CIOQ switches. The key to make a technique practical in this context is that, to make the load balancing, input line cards only use *local* line card information. We test our algorithms showing their effects in long-lived TCP flows.

According to our results, TCP flows at a rate below 250 times the lightpath capacity do not significantly suffer from packet reordering problems, whatever technique is used (10 Mbps in lightpaths at 2.5 Gbps). This is because consecutive packets of the flow are naturally separated. For TCP flows with higher rates, it becomes evident that a wrong choice of the AS technique can greatly degrade the performance of long-lived TCP flows. Fortunately, one of the proposed techniques, which we call *effective*-JSQ (e-JSQ) is able to eliminate in practice the packet out-of-sequence problem in most of the cases, and reduce it to a reasonable level in the rest. Then, all the network traffic, including TCP, can enjoy of the benefits of LB+AS without throughput penalties caused by packet reordering.

The rest of the paper is organized as follows. In Section II, we expose the context of packet reordering problem in LB+AS networks. In Section III we present the AS techniques proposed, while Section IV describes the tests conducted on them. Finally, Section V concludes the paper.

II. THE PACKET REORDERING PROBLEM IN LB+AS

A. Impact of packet reordering on TCP

TCP is designed for transporting *elastic* traffic, in the sense that, if the application has an infinite amount of traffic to transmit, TCP adapts its rate by increasing the transmission window as much as it can. When the TCP source *estimates* that a packet was lost, it assumes the network is congested, and reduces the transmission window size, usually halving it. The particular criteria of the TCP source to estimate when packet losses occur and how the transmission window size reacts, varies in different TCP versions.

A TCP technique called *Fast Retransmit* is responsible for making packet reordering a TCP-critical problem. The technique works as follows. When a packet arrives out-of-sequence to the TCP receiver, it sends an ACK with the same sequence number as the one before (since the received packet does not advance the sequence number). This is called a DUP-ACK. If the TCP source receives three or more consecutive DUP-ACKs it *assumes* that a single packet was lost, causing the transmission window to downsize. As shown in Fig. 1, this reduces the TCP throughput accordingly.

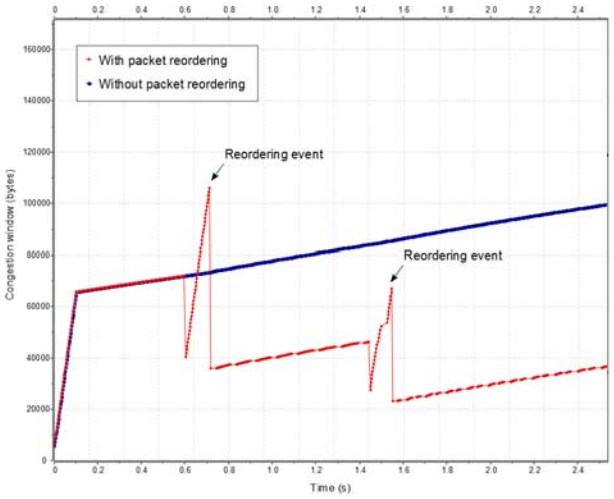

Fig. 1. Evolution of the congestion window of a TCP source comparing whether or not there is packet reordering.

B. Commercial CIOQ routers in a nutshell

In our previous works investigating the LB+AS potential benefits, we assumed that the electronic switching nodes preserved the end-to-end packet sequence. This was based on two assumptions:

1. The router is able to emulate an ideal output queuing (OQ) behavior and packets are forwarded in a cut-through fashion.
2. The implemented AS policy is an ideal JSQ.

In OQ switches, arriving packets are immediately switched to its desired output port, where are buffered waiting for its turn to be transmitted. Although OQ architectures have the advantage of maximizing the throughput of the router, they exhibit a major issue: switch fabric and output queues must operate N times faster than the line rate, where N is the number of input ports (speedup N). Thus, building OQ switches is technologically prohibitive in core routers at high-speed line rates. In contrast, commercial routers use alternative solutions such as combined input-output-queued (CIOQ) architectures (see Fig. 2), which represent a trade-off between OQ switches and input-queued (IQ) switches, which suffer from the head-of-line blocking problem [7]. CIOQ switches implement small buffers at the input (known as virtual output queues, VOQs) and large buffers at the output. Arriving packets are stored in the input line cards in the VOQ corresponding to the desired output port. The switch fabric forwards the packets from the input line cards to the output line cards. A scheduler arbitrates which input line card is granted to send a packet to which output line card. Switch fabrics usually have a small speedup value (typically 2-4). This permits CIOQ switches to have a throughput similar to

that obtained by OQ switches. Also, because of this speed up, the memories in the input line cards are usually empty (e.g. tens of packets), since packets are quickly forwarded to the output line cards. Packets spend most of the time at the output line card buffers, which are prepared to store millions of packets (i.e. buffers of 128 MB in Cisco 12000 GSR).

Fig. 2 depicts the structure of a CIOQ switch which will serve as our reference architecture. When the packets arrive at their ingress line cards, a forwarding decision (the packet output port) is taken observing the packet header, and looking up a cached copy of the routing tables. Then, a segmenter chops the packet into fixed-sized cells, which are subsequently stored in the VOQ corresponding to the packet output port. Cells include a certain overhead and padding for internal switch fabric operation. At the moment decided by the scheduler, cells are transferred to the output line card across the switch fabric. There, they are put into a reassembly queue, waiting for the rest of cells belonging to the same packet. This is needed since cells from different inputs may be interleaved at the same output [8]. Finally, once packets are reassembled they go to the output queue for transmission. This latter queue can be internally arranged in several queues per QoS class.

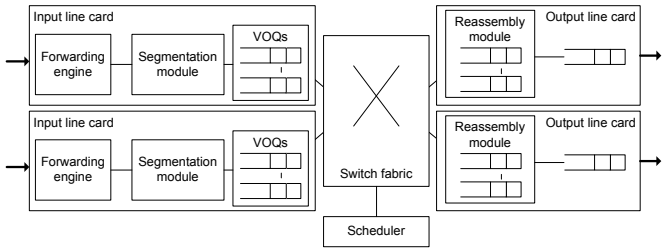

Fig. 2. Architecture of a CIOQ router.

III. IMPLEMENTING AS IN CIOQ SWITCHES

In commercial routers, switching fabric and scheduler circuits are usually accommodated in a single rack, different to those which host the line cards [9]. Racks are connected by optical links, and there is a non-negligible amount of time for communication control. Hence, designing *practical* AS techniques requires in essence that each input line card uses only *local* information (managed and updated in that line card) when making the load balancing decisions.

The first approach for designing AS algorithms is letting the forwarding engine make the decision on the output port of the packet, *at the moment of packet arrival*. Then, the packet is chopped, stored in the appropriate VOQ etc. following the standard switch operation.

During the investigations carried out, that we are presenting in this paper, we chronologically started testing several simple techniques based on (i) letting the forwarding engine take the AS decision using local information, (ii) so that afterwards the packet is chopped, stored in the appropriate VOQ etc. following the standard switch operation.

In this context, the techniques investigated are:

1. Random. Each packet is randomly distributed between the VOQs belonging to the output bundle.
2. Join the Shortest Virtual Output Queue (JSVOQ). Each packet goes to the VOQ with the lowest occupancy (among the ones corresponding to the bundle).
3. Round Robin (RR). Classical circular round-robin among the VOQs of the output bundle: first packet to the first VOQ, next to the second… when the last VOQ receives a packet, next one goes to the first.

As will be shown in the results section, all the previous techniques created large goodput reduction problems in TCP. The reason is that, although all the techniques tend to balance the traffic very well in the bundles in a macroscopic (statistical) view, all of them created frequent short-scale output queue misbalances and subsequent packet reorderings. Fig. 3 helps us to illustrate this. It shows a small network of two CIOQ nodes connected by a bundle of two lightpaths. We only refer to JSVOQ and RR cases.

In the JSVOQ case (Fig. 3a), consecutive packets would be easily reordered due to the fact that the occupancy in VOQs is not representative of the occupation in the buffer of the output line card. Actually, VOQ occupation is usually minimal [6] (in our simulation, lower than two or three full-sized packets), and fluctuates rapidly, so this technique roughly performs as a random policy.

Fig. 3b helps us to illustrate the reasons for packet reordering in the RR case. The RR pointer directs first packet to the VOQ with more pending traffic, which also corresponds to the output queue with more traffic. As a consequence, packet 2 overtakes packet 1. The reason is that, since packets are of different sizes and the scheduler decisions cannot be controlled, the RR pointer can point to an output port that is far from being the optimal JSQ choice.

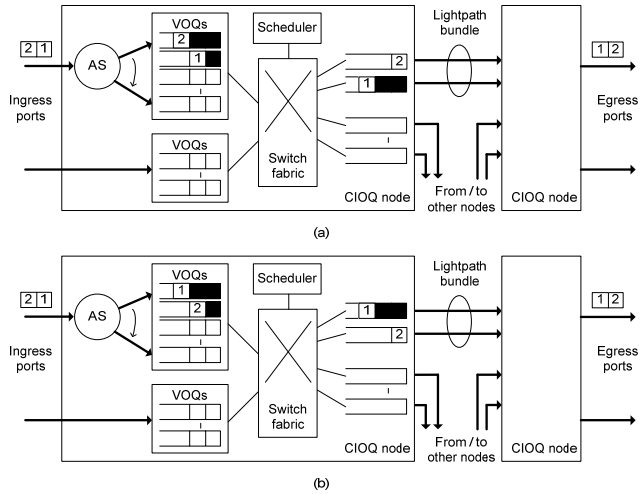

Fig. 3. Example of packet reordering in CIOQ routers when different AS approaches are implemented: (a) Join the Shortest VOQ, (b) Round Robin.

In [10], authors guess that solutions to this problem might be found, but they are deemed to be costly and complex.

However, in next paragraphs we present the e-JSQ (effective JSQ), a different approach to the problem which has shown much better results. Two key ideas are behind e-JSQ:

- *Output port decision*: Upon packet arrival, the forwarding engine identifies the output bundle of the packet. Then, it decides the precise output port in the bundle to forward the packet using a local estimation of the output queue occupancy. In particular, the forwarding engine updates for each output port, a counter of the number of bytes that *this line card* has historically forwarded there. The nickname of *effective* reflects that we only sum the original packet length, without considering cell headers or any padding added internally by the switch fabric. Then, for each new packet, selects the output port in the bundle with the smallest counter (ties are broken randomly). This enforces a perfect balance of the traffic in the bundles. However, note that this differs from an ideal-JSQ, where counters would have an updated sum of all the traffic volume forwarded by *all* the input line cards.
- *One VOQ per bundle*: After the forwarding decision is taken, the packet is segmented and stored. The critical point is that *one single VOQ* stores all the arriving packets targeted to each output bundle. We refer to them as AS-VOQs.

When the first cell of a packet reaches the head of the AS-VOQ, the input line card requests the scheduler transmitting the packet to the desired output port, and no request is sent to other output ports in the same bundle. Naturally, all the (consecutive) cells of the head-of-line packet request the same desired output port. The particular moment in which cells are forwarded depends on the scheduler arbitration.

Note that implementing e-JSQ demands some simple extra process in forwarding engine, minimal changes in the VOQ arrangement, and *no changes in the scheduler operation*. It is the role of the input line card to request to the scheduler the output ports chosen by the AS-VOQ head-of-line packets. But this is transparent to the scheduler. Then, this approach can be applied to any scheduler algorithm, and no changes in the hardware implementing the scheduler are needed, in contrast with the high-cost and complex solutions predicted for input-buffered switches [10].

IV. Simulation Study

In order to assess the different AS techniques proposed and how they affect reorder-sensitive traffic like TCP, we have conducted a series of simulations using the OMNeT++ [11] framework, and the INET modules for simulating TCP/IP networks. The IP routing model in INET has been extended to support anycast switching as described in previous sections.

A. Scenario

Fig. 4 illustrates our simulation setup. We have a three-layered hierarchical model: access layer, metro layer and core layer. Flows from sources go through individual access links, are multiplexed onto metro links, and finally go through backbone links to their destination on the other side of the network. The access layer consists of a set of sub-networks containing N self-similar IP sources and N IP sinks each one, modeling end-hosts. In addition, in the first sub-network of each side a TCP source and a TCP sink are attached. The core layer consists of two nodes connected by a bundle of b_D (bundling degree) lightpaths; the buffer size per input/output port is set to 128 MB. The metro layer consists of a set of aggregation nodes, serving as an access-core gateway; large buffers of 128 MB are used also at the multiplexing nodes to prevent excessive drops at these points. All links are assumed to be bidirectional and run at 2.5 Gbps, except those connecting TCP hosts, which run according to a given U parameter (measured in Mbps). The average physical RTT between each TCP source-sink pair is random, uniformly picked from the interval 15-25 ms (with an average of 20 ms); these small variations in *RTT* are sufficient to prevent synchronization; the propagation delay among IP source-sink pairs is assumed to be negligible.

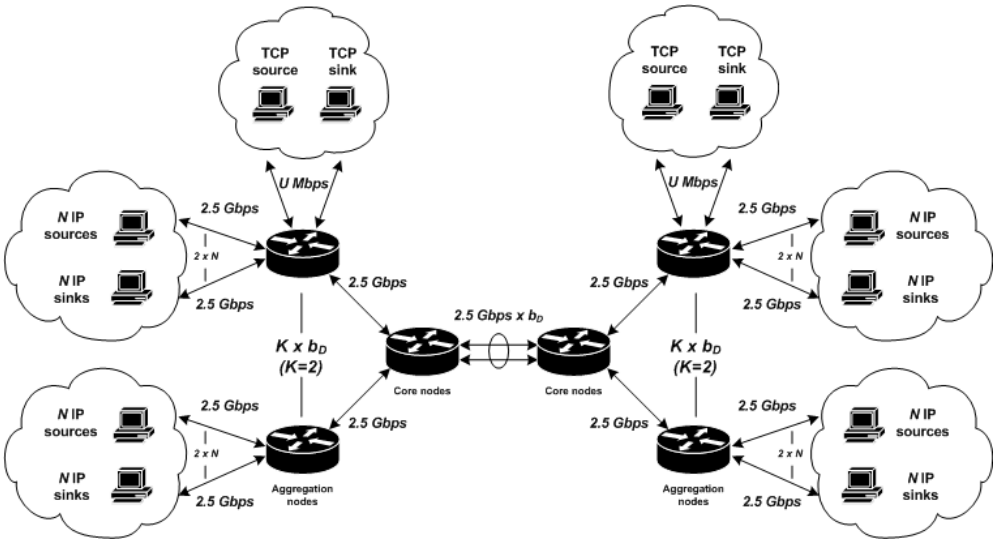

Fig. 4. Simulated network topology.

Each source in one side (left or right) opens a connection to a sink in the other side. The data in each connection goes from the source to the sink, and for TCP connections acknowledgements follow the opposite direction. Since we have sources and sinks in each of the $K \times b_D$ sub-networks, in the left and in the right hand side, data and TCP acknowledgements share the links. The K parameter is used to allow self-similar sources to have long packet bursts not limited by the access link. We use K=2.

The IP self-similar sources are given by the aggregation of 100 ingress ON-OFF sources operating at a peak bit rate of 2.5 Gbps. The load per source is adjusted so that they generate a target load in the backbone links. Each source generates during its ON period IP packets with a length given by a trimodal distribution (40, 576 and 1500 bytes, with probabilities 58%, 33% and 9%). ON periods follow a Pareto distribution of average duration 1 MB, and a shape parameter dimensioned according to [12], so that the superposition of the sources produces a self-similar traffic of Hurst parameter H=0.6. The duration of the OFF period follows an exponential distribution, with average duration adjusted to fit the appropriate load of the source.

The TCP sources transmit as much data as possible. The rate is only limited by the congestion control algorithm used by TCP NewReno (window scaling option enabled) and their access rate U. Data packets have a maximum transfer unit of 1500 bytes.

We order the sources as follows: sources 1 to N are the ones in the upper sub-network in Fig. 4, sources N+1 to 2N are the ones in the sub-network below it, etc. Same ordering applies to the sinks and sources in both left and right sides. Then, the first source in the left side opens a connection with a sink in the first sub-network of the other side, next source with a sink in the second sub-network and so on. Connections from sources in the right side to sinks in the left side are arranged in the same manner. Since in our tests $N=K \times b_D$, it happens that (i) every sub-network has traffic targeted to every other sub-network, and (ii) traffic averages are symmetric from left-to-right and right-to-left, both for self-similar and TCP flows.

Concerning the routing at the core, when LB+AS is not applied, the core nodes see each lightpath in the bundle as an individual link. The routers take a routing decision according to the destination sub-network of the packet. The routing tables are arranged in this case so that each lightpath in the core transmits packets coming from every sub-network. When LB+AS is applied, the routers see the b_D lightpaths as a single link and, in fact, the core routers can aggregate all the IP routes in one. The traffic is spread in the lightpaths according to the per-packet granularity balance enforced by the AS policy. As stated before, we use the CIOQ described in Section II as our reference scheme for core nodes. The aggregation nodes are assumed to be output-queued nodes.

B. Results

We perform our simulations by varying four parameters: (i) the bundling degree (b_D = {2, 3, 4}), (ii) the AS policy (ideal-JSQ, RR, JSVOQ, e-JSQ)[1], (iii) the load on the lightpaths due to self-similar sources (ρ = {0.5, 0.8}), and (iv) the access rate of TCP hosts (U = {10, 100, 1000} *Mbps*). The scheduling algorithm used by the core CIOQ switches is iSLIP [13] with a speedup of 4, and packets are segmented before being enqueued at the VOQs, into fixed-sized cells of 64 bytes (48 payload bytes, padding-filled when necessary). Cells are reassembled into packets at the output line cards, prior the output FIFO queues. Each experiment runs until one TCP sink receives successfully a file of a given size depending on the access rate U: (i) U = 10 Mbps, file size = 10 MB; (ii) U = 100 Mbps, file size = 250 MB; and (iii) U = 1000 Mbps, file size = 500 MB. Apart from the AS techniques tested, we also execute the simulations considering a network where LB+AS is not applied, and CIOQ switches have a standard operation.

The performance metrics measured are: (i) average queuing time in the output ports of the core routers, (ii) average queue length of those ports, and (iii) average goodput of TCP connections. For space reasons, we only show detailed results for TCP goodput, but we briefly comment about every metric.

1) User-centric performance: TCP goodput

In Tables 1-3 the goodput results of TCP connections are shown. Results reveal that when naïve AS schemes (RR, JSVOQ) are applied, the goodput is reduced significantly due to packet reordering. The exception is for a TCP peak rate of U=10 Mbps: when the access rate is too much lower than the core rate, bursts are naturally spread out, so packets from the same connections are separated enough to avoid reordering. The case U=10 Mbps, ρ=0.8 still shows some goodput reduction, since higher loads means more traffic variability and some occasional fast-retransmit episodes.

Fortunately, when e-JSQ is applied the goodput increases and reaches in most cases the values obtained by ideal-JSQs, where there is no reordering at all. Note that, in general, this means a goodput significantly better than when LB+AS is not applied in the network. For higher values of the access rate, and higher values of the number of lightpaths bundled, e-JSQ behaves worse. For instance, when the access rate is high (U=1 Gbps) and TCP peak rate is in the same order of magnitude of the lightpath bit rate, a noticeable goodput reduction occurs in e-JSQ respect the ideal case. However, it is interesting to note that this goodput reduction comes from very infrequent *Fast Retransmit* episodes (e.g. 5-10 events during the transmission of a 500 MB file). The point is that TCP NewReno, and the rest of classical TCP flavors, grows the congestion window slowly: one packet per RTT (due to the additive increase, multiplicative decrease mechanism). Hence, the time to recover from a reordering is very high. Current TCP implementations such as CUBIC (in Linux kernel from 2.6.13) or Compound TCP (used in Microsoft Windows OS from Vista), both appeared in 2006, use other window growing mechanisms which mitigate the under-utilization of the network. Unfortunately, we could not test these TCP versions yet, since they are not currently included in the OMNeT++ INET framework.

[1] Results for the Random AS case are not shown since their performance is worse than the one of RR and JSVOQ algorithms.

Under the light of these results, and observing the very infrequent *Fast Retransmit* episodes, we conjecture that e-JSQ eliminates in practice, and for most of the network users, the packet reordering issue. Users interested in very high-bandwidth and long-distance best-effort connections would not use TCP protocol (or at least TCP NewReno), or would maybe split the data into several lower bandwidth TCP connections. Moreover, as lightpath bit rates grow, TCP connections with higher peak rate would decrease its packet reordering, since packets of the same flow would be interleaved by more packets of other flows. Then, results shown here should be seen as a pessimistic lower bound to reordering performance with 10/40/100 Gbps lightpaths.

TABLE I. AVERAGE TCP GOODPUT IN MBPS (U = 10 MBPS)

	ρ = 50%			ρ = 80%		
Architecture \ b_D	***2***	***3***	***4***	***2***	***3***	***4***
Not LB+AS	9.6	9.7	9.7	9.3	9.4	9.0
Ideal-JSQ	9.7	9.7	9.7	9.6	9.7	9.7
e-JSQ	9.7	9.7	9.7	9.6	9.6	9.6
RR	9.6	9.5	9.6	8.2	7.9	8.3
JSVOQ	9.4	9.5	9.5	6.8	7.7	8.1

TABLE II. AVERAGE TCP GOODPUT IN MBPS (U = 100 MBPS)

	ρ = 50%			ρ = 80%		
Architecture \ b_D	***2***	***3***	***4***	***2***	***3***	***4***
Not LB+AS	81.6	85.9	83.0	57.8	66.5	66.7
Ideal-JSQ	90.1	91.4	90.2	71.1	82.3	85.5
e-JSQ	90.1	90.2	86.6	57.4	48.7	40.6
RR	23.7	29.6	33.1	11.0	6.1	33.2
JSVOQ	21.5	25.8	27.1	5.7	8.7	32.1

TABLE III. AVERAGE TCP GOODPUT IN MBPS (U = 1000 MBPS)

	ρ = 50%			ρ = 80%		
Architecture \ b_D	***2***	***3***	***4***	***2***	***3***	***4***
Not LB+AS	181.2	195.4	187.7	100.6	119.2	112.5
Ideal-JSQ	220.0	220.9	215.3	135.2	166.4	183.2
e-JSQ	219.0	149.9	169.3	120.8	59.5	69.8
RR	183.0	137.8	148.4	26.4	16.6	26.8
JSVOQ	176.2	120.2	131.4	6.9	103.0	132.6

2) Operator-centric performance: buffer occupation

Similarly as in [2][5], we observe that application of LB+AS brings an strong reduction of the average queuing times and the average queue sizes in the routers, between one and two orders of magnitude in most cases (better with more lightpaths bundled). As an example, average queuing delay drops from the order of 10 ms to 0.1-10 ms for networks loaded at a ρ=50%, and from about 30 ms to 2.5 to 10 ms in networks loaded at a ρ=80%.

V. CONCLUSIONS

The studies in [1][2][5] have shown that the application of LB+AS can increase the economic efficiency in optical networks, without the need of any upgrade in the optical network equipment. These works were based on ideal implementations of AS techniques, which avoided packet reordering. Thanks to this, the statistical multiplexing gain granted by the per-packet traffic balancing, was not degraded in packet-order-sensitive traffic like TCP.

In this paper, we investigate for the first time on AS schemes which permit a practical implementation in high-speed, input-buffered switches. Several AS algorithms are presented. Their performance is tested with traffic containing packet-reorder sensitive flows like long-lived TCP connections. Results show that, if AS techniques are not designed carefully, TCP goodput can be greatly degraded. Among the algorithms presented, e-JSQ emerges as a very interesting option which: (i) allows a simple implementation in high-speed switches (without the need of changing the scheduler), (ii) makes the packet reordering negligible in practice, for the normal use of the network. Furthermore, packet reordering is expected to decrease more and more, as lightpath bit rates increase. In authors opinion, the results in this paper, together with those obtained in previous works, put the focus on LB+AS as a strong combination to be implemented in today optical networks.

REFERENCES

[1] P. Pavon-Marino, "Lightpath bundling and anycast switching: a good team for multilayer optical networks," in *Proceedings of the 15th Conference on Optical Network Design and Modeling* (ONDM 2011), February 2011.

[2] P. Pavon-Marino and J.L. Izquierdo-Zaragoza, "Lightpath bundling and anycast switching (LB+AS): a new paradigm for multilayer optical networks," *IEEE Communications Magazine*, vol. 50, no. 8, pp. 89-95, August 2012.

[3] C. Hopps, "Analysis of an Equal-Cost Multi-Path Algorithm," *Internet RFC 2992*, November 2000.

[4] S. Kandula, D. Katabi, S. Sinha, and A. Berger, "Dynamic load balancing without packet reordering," *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 2, pp. 51-62, April 2007.

[5] P. Pavon-Marino and J.L. Izquierdo-Zaragoza, "TCP performance in an optical link applying lightpath bundling and anycast switching techniques," in *Proceedings of the 14th International Conference on Transparent Optical Networks* (ICTON 2012), July 2012.

[6] N. Behesti *et al.*, "Optical Packet Buffers for Backbone Internet Routers," *IEEE/ACM Transactions on Networking*, vol. 18, no. 5, pp. 1599-1608, October 2010.

[7] M. Karol, M. Hluchyj, and S. Morgan, "Input versus output queueing on a space division switch," *IEEE Transactions on Communications*, vol. 35, no. 12, pp. 1347-1356, December 1987.

[8] M. Ajmone-Marsan, A. Bianco, P. Giaccone, E. Leonardi, and F. Neri, "Packet Scheduling in Input-Queued Cell-Based Switches," *IEEE/ACM Transactions on Networking*, vol. 10, no. 5, pp. 666-678, October 2002.

[9] I. Keslassy *et al.*, "Scaling internet routers using optics," in *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications* (SIGCOMM '03), August 2009.

[10] J.C.R. Bennett, C. Partridge, and N. Shectman, "Packet Reordering is Not Pathological Network Behavior," *IEEE/ACM Transactions on Networking*, vol. 7, no. 6, pp. 789-798, December 1999.

[11] A. Varga, "The OMNeT++ Discrete Event Simulation System," in *Proceedings of the European Simulation Multiconference* (ESM'2001), June 2001.

[12] W. Willinger, M. S. Taqqu, R. Sherman, and D. V. Wilson, "Self-Similarity Through High-Variability: Statistical Analysis of Ethernet LAN Traffic at the Source Level," *IEEE/ACM Transactions on Networking*, vol. 5, no. 1, pp. 71-86, February 1997.

[13] N. McKeown, "The iSLIP Scheduling Algorithm for Input-Queued Switches," *IEEE/ACM Transactions on Networking*, vol. 7, no. 2, pp. 188-201, April 1999.