TCP Congestion Control in Datacenter Optical Packet Network on Hybrid Switches

Artur Minakhmetov, Cédric Ware, and Luigi Iannone
LTCI, Télécom ParisTech, Université Paris-Saclay
Paris 75013, France
artur.minakhmetov@telecom-paristech.fr

***Abstract*—Optical Packet Switching (OPS) has long promised performance and energy consumption improvements by doing away with optical-to-electronic conversions required by electronic packet switching (EPS); however, not having practical optical buffers makes OPS highly vulnerable to contention.**

This study reports on the possible and plausible use of OPS technology on datacenter networks by coupling two concepts: optical switches with shared electronic buffers, also known as hybrid switches; and introduction of TCP congestion control algorithms (CCAs) to control transport of optical packets. The Stop-And-Wait (SAW) and the modified Additive Increase Multiple Decrease (mAIMD) CCAs families are reviewed. For SAW, a basic version and a modified one – Stop-And-Wait-Longer (SAWL), adapted for hybrid switches, are analyzed. As for mAIMD, a TCP Selective Acknowledgment (SACK) implementation and its simplified modification TCP mSACK are studied. It is successfully shown that these algorithms paired with the use of shared electronic buffers in hybrid switches significantly outperform bufferless all-optical switches and reach the level of all-electronic switches in Datacenters and Local Area Networks (LAN) in terms of network throughput.

***Index Terms*—Packet-switched networks, Packet Switching, All-optical networks, Optical Switches, Hybrid Switches, TCP, Congestion control, CCA**

I. Introduction

PACKET switching is at the heart of current data networks due to its high flexibility and efficient use of available capacity through statistical multiplexing. However, switching currently must be performed electronically, despite the fact that most of the traffic is transmitted through optical signals. This incurs many optics-to-electronics-to-optics (OEO) conversions, thus a cost in terms of energy and performance bottlenecks of the electronics. Given the traffic's exponential growth, this cost leads to an unsustainable increase of energy consumption and other operational expenses. Optical packet switching (OPS) initially has been proposed in 1990s in [1], [2] and gained its maximum interest in mid-2000s [3]. However, with traffic being asynchronous and in the absence of a technology that would make optical buffers in switches a reality, the contention issue rises, leading to poor performance in terms of Packet Loss Ratio (PLR) [4], thus making the OPS concept impractical. To the present moment, several solutions have been proposed to bring the OPS technology to functional level [5], among which we shall focus on two approaches: hybrid switches, and special TCP Congestion Control Algorithms (CCA).

First, the idea of a hybrid switch consists of coupling an all-optical bufferless switch with an electronic buffer [6]: when contention occurs on two (or more) packets, i.e. when a packet requires to use an output port that is busy transmitting another packet, it is switched to a shared electronic buffer through Optical-Electrical (OE) conversion. When the destination output is released, the buffered packet is emitted from the buffer passing Electrical-Optical (EO) conversion. However, in the absence of contention (which is the case for most packets), the hybrid switch works as an all-optical switch, without any wasteful OE and EO conversions, offering the possible cut-through operation. Adding a shared buffer with only a few input-output ports lets us considerably decrease PLR compared to an all-optical switch [7], and bring its performance to the one of electronic switches, but now with an important reduction in energy consumption. One would save the OEO conversions for the most packets: Samoud et. al. [8] show that for a hybrid switch, that possesses roughly half as many inputs (30) for electronic buffers than switch inputs (64), one can gain more than 50 % in reduction of OEO conversion in the worst case of 100% load, compared to an all-electronic switch.

Alternatively, Argibay-Losada et. al. [9] propose to use all-optical switches in OPS networks along with special TCP CCAs, in order to bring the OPS network throughput up to the same levels as in Electronic Packet Switching (EPS) networks with conventional all-electronic switches, negating the effect of the poor PLR of a standalone all-optical switch and ensuring high levels of quality of service of the whole network. In protocol design one could bring forward two main aspects: properly setting the Retransmission Time-Out (RTO) and the size of the congestion window, both of which must be properly set with initialization of every TCP connection and adjusted along its lifetime. RTO is one of the main parameters used to figure out whether we should consider the sent packet as lost and resend it, or keep waiting for the acknowledgement before sending the next packet. When transmission is successful and without losses, RTO is set to a value close to the Round-Trip-Time (RTT), i.e., the time elapsed between the start of sending a packet and reception of the corresponding acknowledgement. The other important aspect of the TCP CCA is the congestion window, i.e. how many packets can be sent before pausing and waiting for acknowledgments. The answer to the question of the evolution of the congestion window depends on the TCP variant and the

conditions of the network. According to authors of [9], in the case of an all-optical OPS network, the TCP Stop-And-Wait (SAW) algorithm, which maintains only one packet in flight, is adapted for datacenters and LAN networks; whereas TCP modified Additive Increase Multiplicative Decrease (mAIMD) is aimed at making OPS work for larger Metropolitan Area Networks (MAN) networks.

In our previous work [10] we reviewed only TCP SAW and its modification called TCP SAWL on hybrid switches in LAN and datacenter networks.

In this paper we provide several new contributions to our previous analysis. We introduce the throughput analysis of TCP mAIMD, specifically their versions of TCP SACK based on [11] and a modified one, named TCP mSACK. Altogether, we review datacenter network performance under TCP SAW, TCP SAWL, TCP SACK and TCP mSACK paired with hybrid switch use. Furthermore, we add a comparison of performance of a network composed of electrical switches, which simulates EPS, the conventional existing solution applied in networks. Based on this analysis, we draw conclusions on the plausibility of using hybrid switches in a datacenter network.

Additionally, in order to compare in a more complete way the case of OPS on hybrid switch versus EPS, we introduce the case,where the number of electronic buffer inputs in a hybrid switch is equal to general number of inputs of the switch. For each optical input port such a switch would have an electronic buffer input, which would be used only in case of packets contention. Cut-through mode of operation would be applicable for general case, and store-and-forward only for contended packets. We call this switch a fully-buffered hybrid switch.

Fully-buffered hybrid switch can be seen as well as an approximation of the high-end cut-through electrical switch. Such hybrid switch offers a cut-through option for non-contended packets, same way as cut-through electrical switch. Contended packets will be buffered in a fully-buffered hybrid switch, almost the same way as in cut-through electrical switch. The difference lies in managing buffered packets: cut-through electronic switch may offer emission to a packet, still undergoing the bufferization process, while hybrid switch would require a completion of bufferization. However, in presence of just two packets in the buffer, requiring the same output (under high load this happens often), even the cut-through electronic switch would be forced to require a completion of packet bufferization, thus negating the difference between buffered packets management. While considering fully-bufferd hybrid switch as an approximation of the high-end cut-through electrical switch, which essentially entails the same performance, we limit ourselves with review of fully-buffered hybrid switch, and do not introduce the cut-through electrical switch. We must note here that fully-buffered hybrid switch entails preferable reduction in OEO conversions, making itself a good candidate for overall energy savings, that cut-through electrical switch misses completely.

In this study we present a quantification of the performance of the hybrid switch in the context of the whole datacenter network and underline the TCP CCAs needs to be applied, that would discover the potential of the hybrid switch. We show

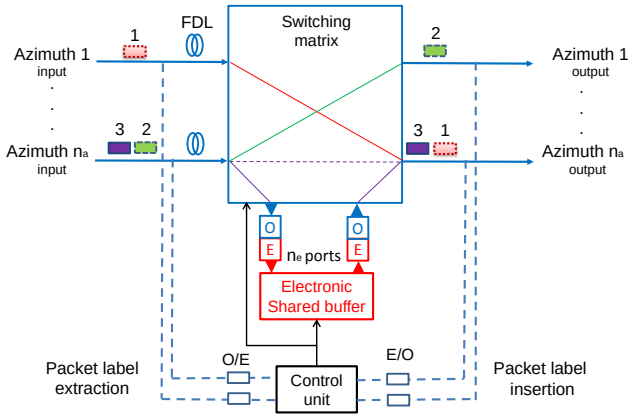

Fig. 1: General architecture of hybrid optical packet switch

that, in general, no matter what TCP CCA is used, networks with hybrid switches outperform networks with all-optical ones in terms of the network throughput. Additionally, in the case of fully-buffered hybrid switches, they reach the same level as those with electronic switches under TCP mSACK, or even slightly outperform EPS network while employing TCP SACK. Furthermore, for the case of hybrid switches we show that TCP SACK significantly surpasses TCP SAW and TCP SAWL in terms of throughput.

The paper is organized as follows: Sec. II lays out the architecture of the hybrid switch we employ, Sec. III outlines TCP CCAs used for this study, Sec. IV presents the simulation conditions, while Sec. V discusses the results obtained and, finally, Sec. VI offers our main conclusions.

II. Hybrid Switch Architecture

In this section we review existing hybrid switch solutions presented in scientific literature, address its general architecture and explain the assumptions made for our study.

Throughout 2010s there was a persistent interest in finding technological solutions for OPS, exploring different enabling components and architectures. In 2010 Xiaohui Ye et al. [12] presented a Datacenter Optical Switch (DOS), an optical packet switch, that could be seen as a prototype of the hybrid one: switching was performed through combination of Arrayed Waveguide Gratings (AWGs) switching matrix with Tunable Wavelength Converters (TWC), which essentially performed OEO along with wavelength packet conversion in order to switch it accordingly through AWGs. Similarly to the hybrid switch discussed in this paper, DOS managed contentions through the shared electronic buffer, storing the contended packets. In 2012 Ryo Takahashi et al. [13] presented a similar concept, called Hybrid Optoelectronic Packet Router (HOPR). HOPR, despite its name, was not exactly a hybrid switch, as performed OEO conversions for all the packets by TWC in order to route them.

In 2013 Y. Yin et al. [14] presented low-latency interconnect optical network switch (LIONS), while basing it on DOS. Leaving the same concept of switching TWC+AWGs authors explored different contention resolutions schemes along with

distributed (channel-specific) electronic buffer, mixed electronic buffer (some buffers are shared, some distributed). As well authors of [14] proposed the use of All-Optical Negative Acknowledgement (AO-NACK), where the LIONS doesn't have any buffers, but sends to servers the AO-NACK on the same channel by back propagation, indicating that the packet should be retransmitted. That architecture could be seen as a TCP related control scheme, however, the reviewed topology is a star, consisting of only one switch interconnecting a lot of servers.

In 2016 T. Segawa et al. [15] proposed an optical packet switch that is much closer to hybrid switch than HOPR or LIONS/DOS: it performs switching of optical packets through broadcast-and-select (B&S) and then re-amplification by semiconductor optical amplifier (SOA). This switch splits the incoming optical packet into several ways corresponding to output ports, blocks those that didn't match the packet's destination, and then re-amplifies passed packet by SOA. Shared electronic buffer solves the contention. The OEO conversion is made only for contended packets, as in discussed in this paper hybrid switch case, while keeping the non-contended packet in optical domain.

In 2017 X. Yu et al. [16] addressed the issue of back-propagating AO-NACK through several hops, what LIONS were missing due to the star topology, and proposed the concept of solving this issue by introducing TWC on output ports of switch (type LIONS/DOS) for back-propagating AO-NACKs, which may seem effective but still adding to OEO conversions. Additionally, this solution was not numerically evaluated.

In 2018 N. Terzenidis et al. [17] proposed a 256x256 optical packet switch, which used a combination of "B&S+SOA" and AWGs scheme, still undertaking OEO conversion for a packet, however using Fiber Delay Lines (FDL) for contention resolution. Offering an impressive number of ports, such solutions still requires OEO conversions and employs FDLs, that introduce fixed delay for a packet, which may be seen as a drawback for asynchronous traffic compared to electronic shared buffer.

After this review we can conclude on interesting solution from [15], the only drawback of which is a switching matrix that induces signal losses and requires re-amplification. However this could be overcame by use of recent developments on switching matrices: for example recent T. Chu et al. [18] fast (several ns) 32x32 switching solution, based on Mach-Zehnder Interferometers (MZI) switches arranged in Benes topology. This switching matrix could be potentially applied in 16x16 fully-buffered hybrid switch.

All of the presented solutions above have some common principle blocks, that we are emulating or approximating in our study in order to approach hybrid switch functions. The general structure of the hybrid switch and its processes are presented in Fig. 1. When a packet enters the switch, it carries along a label containing the destination address. This label is extracted from the communication channel through a splitter (usually 90:10) or a 1x2 MZI switch and then directed to the Control Unit where it undergoes O/E conversion. The Control Unit represents a Field-Programmable Gate Array (FPGA), which controls Switching Matrix and Electronic Shared Buffer. While the Control unit analyzes the label, the packet is delayed in FDLs so to give time to the FPGA to adjust the Switching Matrix. Then, it would either route a packet to the desired output, route and store it in a buffer or finally drop it. If a Control Unit decides to route the packet to desired output or eject the packet from buffer, it will generate new label, performing EO conversion, to add it to a packet on switch's output. This mechanism let us to stay out from OEO conversion of the whole packet.

The Switching matrix could be implemented by the technologies described above: B&S switch + SOA, TWC+AWGs, or even assembled in Benes Architecture multiple MZIs. Electronic Shared buffer is supposed to be implemented by burst receivers.

This study employs following assumptions: label processing, Control Unit, Switching Matrix and Electronic Shared buffer are generic and switching time is negligibly small. However, we pay close attention to the routing processes.

Hybrid switch has n_a inputs and n_a outputs, as shown in Fig. 1, representing non-wavelength-specific input and output channels, or Azimuths. Input/output pair of the same index represent a bidirectional channel, thus making n_a channels for a switch. Another important parameter is n_e: n_e inputs and n_e outputs of a buffer. These are the channels through which packet is routed/emitted to/from a buffer.

The routing algorithm for hybrid switch is the following: a packet enters the switch and checks if required Azimuth output is available. If yes, the packet occupies it. Otherwise the packet checks if any of buffer inputs are available. If yes, it occupies one and start bufferization. If none of the buffer inputs are available, the packet is dropped. If the packet is buffered, it constantly checks if it is first in buffer queue and the required Azimuth output is available; when two of these conditions are met, the packet is emitted from the buffer output and packet occupies the required Azimuth output.

III. TCP CCA DESIGN IN OPS NETWORKS

In this section we will review two families of TCP CCAs that could be seen as interesting to study in an OPS datacenter network under different types of switches: TCP SAW in subsection III-A and TCP mAIMD in subsection III-B.

A. TCP SAW and TCP SAWL

While designing protocols in OPS networks one could mostly recognize two cases: when there is at most one packet in flight, and when there might be several. In the former case it is proven to be efficient to use TCP SAW [9] and its mostly adapted scenario is to use it in datacenters with their relatively short distances. If one decides to use Jumbo Ethernet packet of size 9 kB on 10 Gb/s network interface cards in a datacenter with the longest distance between servers of 600 m, there will be a propagation delay of 2.9 µs[1], which is less than half the 7.2 µs needed to transmit the packet, i.e. packet duration τ. This essentially means that there can

[1]Considering speed of light in fiber with refractive index of $n = 1.45$.

be only one packet in flight, and it's more prudent to wait for the acknowledgement before sending the next packet, or retransmits the current packet when the RTO timer expires. Here an RTO T_1 of 1 ms is taken as the initial value, instead of conventional 1 s as suggested by Paxons et al. in [19]. If the corresponding acknowledgement packet is not received within this time, for the retransmission the RTO is now multiplied by a constant factor $\alpha > 1$ so that the RTO is updated as:

$$T_i = \alpha \cdot T_{i-1}, \tag{1}$$

up to a maximum value of $T_i = 60$ s. When the acknowledgment is received, the RTO is updated to a weighted average of its current value and the measured RTT γ:

$$T_i = \beta \cdot \gamma + (1 - \beta) \cdot T_{i-1}, \tag{2}$$

with $\beta \in (0, 1)$. In our evaluation we used $\alpha = 1.1$ and $\beta = 0.5$ following the more suitable values suggested in previous works [9]. The choice of the RTO timer's initial value can be justified after consideration of: *i)* the absence of buffering delays in the all-optical network, resulting in a low RTT variance, that is only due to queuing delays; *ii)* the RTT in LANs, that is below of T_1 of 1 ms; *iii)* the RTT in MANs, that is in the order of T_1 of 1 ms. Thus, as long as the destination server is not overloaded with connections, there is no point in waiting for the acknowledgment longer than the true RTT, or a value close to it, in our case the initial RTO. This way the timer helps recover from losses fast enough to maintain a high throughput.

We further developed basic version of TCP SAW into a modified one, called TCP SAWL and initially proposed it in [10]. The TCP SAW works efficiently for all-optical switches, but does not allow to take advantage of the buffers. Indeed, even putting/extracting a packet into/from the buffer adds up to the RTT, becoming longer than the RTO estimated for the previous non-buffered packet, and thus the server will consider such packet as lost. To overcome this limitation, we proposed a modification of the SAW algorithm, so that the RTO is increased by a multiple p of packet duration τ, so as to give a chance to packets having traversed up to p buffers to arrive before the RTO, limiting unnecessary retransmissions. Hence, instead of the T_i shown before, we take as RTO:

$$T_i' = T_i + p \cdot \tau. \tag{3}$$

Our simulations consider $p \in \{0, 4\}$, i.e. basic version of TCP SAW, and one that waits for packets buffered four times, as favorable value we found in [10]. The essential modification in SAW algorithm is to wait slightly longer, that is why we are referring to it as Stop-And-Wait-Longer – SAWL.

B. TCP mAIMD in TCP SACK and mSACK implementation

When distances increase, i.e. in the MAN case, the network is usually able to accommodate several packets in flight. Thus, the use of network bandwidth with TCP SAW becomes much less effective with its single packet in flight, and the throughput decreases drastically. To fight this issue one could prefer to use TCP protocols with variable Congestion WiNDow (CWND), measured in bytes, regulating the number of packets in flight, depending on occurred losses. Authors of [9] propose to use an Additive Increase Multiple Decrease (AIMD) algorithm with reduction of the initial RTO towards 1 ms, thus naming it as TCP mAIMD, as a strategy to control the CWND. The general principle of this algorithm is: if an acknowledgment of the packet is received before the RTO timer expires, the CWND is increased linearly, and thus several new packets can be sent; otherwise, when loss is detected, the CWND is decreased by some predefined factor. Such a strategy helps to achieve better use of bandwidth than TCP SAW and increases the throughput in MAN OPS networks.

In this paper we are reviewing the algorithm SACK [11], a candidate for mAIMD, and a modified version of SACK with more aggressive CWND updating. As we review the TCP CCAs together with hybrid switches, TCP SACK is a good candidate for further study as well from point of view of energy consumption.

TCP SACK is chosen for implementation as it allows to receive selective acknowledgments (SACKs), i.e. when one packet of the several sent is lost, the server will acknowledge all the packets that successfully arrived to the destination, thus indicating to the sender the missing packets. TCP SACK is based on TCP Reno, which has three different transmission phases. Initially it uses the "slow start" phase, where CWND increases exponentially in time when no losses occurred till a predefined $ssthresh$ value. Then, in the absence of losses, algorithm enters a "congestion avoidance" phase: CWND increases linearly. A "fast recovery" phase, is used when loss is detected with reception of 3 duplicate acknowledgments (DUP ACK) of the same packet sent, and packets considered to be lost are retransmitted before RTO expiration with halving current CWND, and updating $ssthresh$ [20] as follows:

$$ssthresh_i = \max\left(\frac{\text{CWND}_{i-1}}{2},\ 2 \times \text{SMSS}\right), \tag{4}$$

$$\text{CWND}_i = ssthresh_i, \tag{5}$$

where SMSS stands for Sender Maximum Segment Size and defines the maximum useful payload that a packet can carry. During "fast recovery" TCP helps recover from losses, and doesn't change CWND. However, if the RTO timer ever expires, CWND is set to 1 packet in flight.

TCP SACK by [11] has another phase defined as "rescue retransmission", which should come during the "fast recovery" phase and allow to retransmit one packet that is not considered as lost. Nevertheless, we opt to omit such phase in our implementation, in order to limit the impact of non-essential retransmissions on network load. As in TCP mAIMD, the initial RTO is reduced to 1 ms. We must note here, that in order to make a true comparison to existing electronic switch solutions in datacenters using different versions of TCP AIMD, for the case with electronic switch we must review not only reduced initial RTO version, but a conventional initial RTO of 1 s.

We present in Fig. 2a the example of possible evolution of CWND during various phases for TCP SACK depended from valid ACK reception instant, i.e. CWND change instant, so exponential growth of the CWND during "slow start" in time appears to be linear. We observe the evolution of the CWND

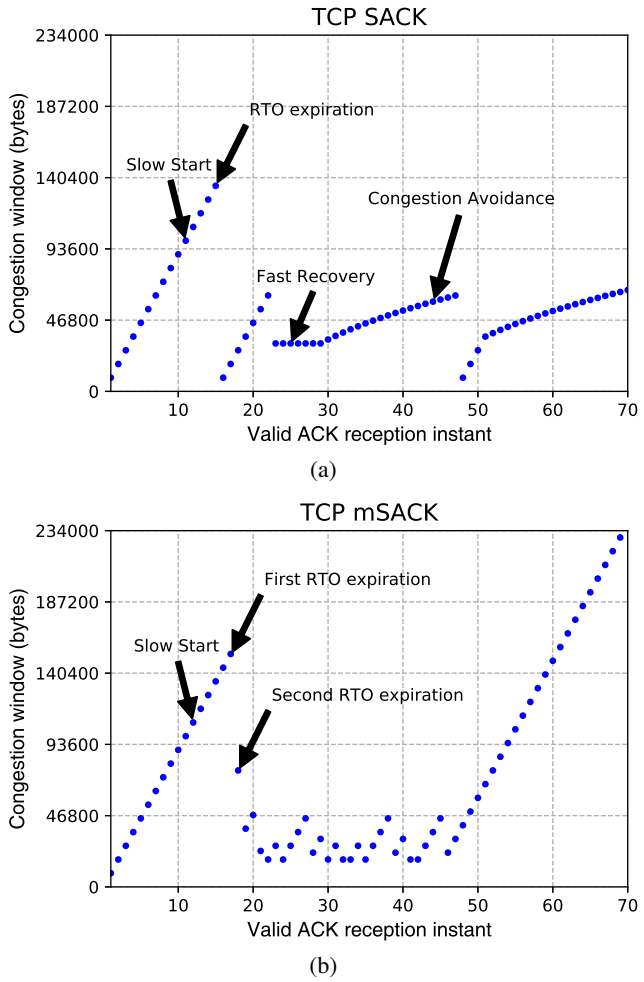

Fig. 2: Example of Congestion Window evolution for: a) TCP SACK, b) TCP mSACK.

under the losses induced by network operation of topology defined in Sec. IV with link length of 100 m consisted of hybrid switches with 5 input/output electrical ports. In our simulations CWND was measured in bytes and is used to determine how many packet of the predefined size could be sent.

As we can witness in Fig. 2a, TCP SACK retains all the phases described above, it starts with "slow start", with first RTO expiration CWND is decreased to the size of one packet, or SMSS. "Fast recovery" phase starts from reception of 3 DUP ACKs, $ssthresh$ is set as half of previous CWND, as well as new CWND, which is maintained till the end of this phase. As it ends with $CWND = ssthresh$, the algorithm enters in "congestion avoidance" phase, after that we observe again RTO expiration, slow start phase and, eventually, transmission ends on congestion avoidance step.

The modified algorithm TCP SACK, or TCP mSACK, has a more aggressive CWND variant than TCP SACK, without "congestion avoidance" and "fast recovery" phases, which means CWND increases exponentially all the time as long as losses don't occur; also, upon RTO expiration CWND is not reduced to the size of 1 packet in flight, but halved. Receiving 3 DUP ACK doesn't lead to the consideration of packet lost and doesn't decrease CWND. This CCA has the potential in being adapted more to MAN, than to data center networks, as long server-to-server links in MAN has the capacity to fill themselves with a lot of packets. Taking into account aggressive nature of mSACK, it may fill such links faster, than SACK, leading to better throughput. Such hypotheses will be testes in our next studies. However, to make further analysis complete, we include this CCA in this review, and complete its analysis with the EPS case and the fully-buffered hybrid switch.

Considering case of TCP mSACK in Fig. 2b, under same conditions of network parameters as in TCP SACK case in Fig. 2a, we can see, that only RTO expiration changes the CWND, and no "congestion avoidance" or "fast recovery" exists. We bring attention of the reader that direct quantitative comparison of TCP SACK and TCP mSACK cases in Fig. 2 is not applicable, on the contrary to the qualitative comparison that was made earlier, as each case undergoes different number of packet losses at different instants. These packet losses depends on the whole network operation with hundreds connections in it in parallel, and even with all other equal parameters of the network, connections under different CCAs will differently influence other connections, inducing losses at different instances.

As for the case of TCP SAW and TCP SAWL, the congestion window will be always constant and is given by

$$CWND_i = SMSS, \tag{6}$$

as it maintains only one packet in flight.

IV. Experimental Simulations

We simulate the communications of datacenter servers by means of optical packets, for three scenarios: *i)* when the network is composed of only all-optical switches, *ii)* when it is composed only of hybrid switches and *iii)* when it is composed only of conventional store-and-forward electronic switches. Case *iii)* represent a switch that undertakes whole OE and EO conversions and receives a whole packet, until sending it further.

Communications consist of transmitting files between server pairs through TCP connections. The files' size is random, following a lognormal-like distribution [21], which has two modes around 10 MB and 1 GB.

File transmission is done by data packets of Maximum Transmission Unit (MTU) size[2], i.e., 9 kB. This value defines packet's payload and corresponds to Jumbo Ethernet frame's payload. We choose this value so as to be related to previous research on all-optical packet switched networks [9]. A priori the size of the packet will influence the throughput, however we choose the 9 kB to make favorable conditions [22] for all-optical network case while using TCP SAW, and compare its best performance with hybrid and electrical switches.

In our study we also use SYN, FIN, and ACK signaling packets. We choose for them to have the minimal size of the Ethernet frame of 64 bytes [23], and assume that they do not carry any data related to the file content (payload).

[2]In our case it defines SMSS as well.

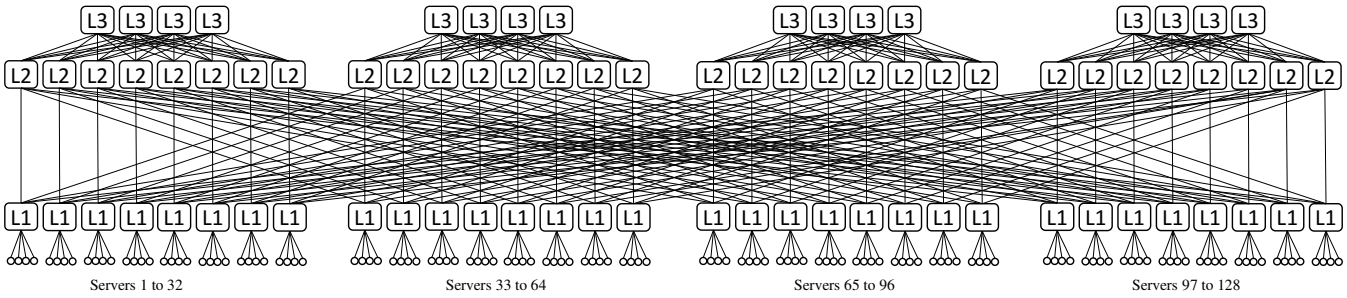

Fig. 3: Fat-tree topology network, interconnecting 128 servers with three layers of switches.

These packets could carry payload, but in our case they don't, as in the context of the same TCP connection (as it's shown further) the destination server uniquely accepts a file, thus not sending any payload back to the source server. The same destination and source servers could change their roles, and start sending a file in opposite direction, but this will be regulated by separate TCP connection. As file transmissions demands (and thus TCP connection demands) arrive independently (cf. Poissonian process further), we can't consider jointly the parallel transmissions of two files in two opposite directions, even between the same pair of servers. We assume that this minimal size would contain all the relevant information about Ethernet, TCP and IP layers, carrying the MAC addresses, TCP flags, Seq and Ack numbers, that are necessary for TCP CCA.

As we still need to attach to the MTU all the information about Ethernet, TCP and IP layers, for simplicity, we just attach to these 9 kB a header containing 64 Bytes discussed previously. Thus we are constructing the 9064 Bytes data packet to be used in our simulations, with a duration τ dependent on the bit-rate.

The last data packet of each connection may be smaller than 9 kB in terms of payload, since file size is not an exact multiple of the MTU. The actual transmission of each data packet is regulated by the TCP CCA, which decides whether to send the next packet or to retransmit a not-acknowledged one. To be realistic, the initial 3-way handshake and 3-way connection termination are also simulated. The network is characterized by the network throughput (in Gb/s) as a function of the arrival rate of new connections.

We developed a discrete-event network simulator based on an earlier hybrid switch simulator [7], extended so as to handle whole networks and include TCP emulation. The simulated network consists of hybrid switches with the following architecture: each has n_a azimuths, representing the number of input as well as output optical ports, and n_e input/output ports to the electronic buffer, as shown in Fig. 1. The case of the bufferless all-optical switch corresponds to $n_e = 0$. When a packet is switched to an available azimuth, it occupies it. If the azimuth is busy, then the packet is redirected to the electronic buffer through an electronic port. The packet will then be re-emitted when the output azimuth it needs is released. The re-emission queuing strategy of the buffer is First-In-First-Out (FIFO) for a given azimuth.

In all the cases we consider that the size of the buffer, in terms of Bytes it can hold, is not limited. Samoud et al. in [8] indicated that buffer of a hybrid switch is used only by a few dozens of packets, implying less than 1 MB for a buffer size. However, we notice that maximum buffer size depends on CCA. If we measure maximum buffer size occurred during a simulation, and then take average among all the random seeds, in the worst case for hybrid switches we obtain next results for: SAW – 0.14 MB, SAWL – 0.15 MB, SACK – 1 MB and mSACK 5.5 MB.

Electronic switches have a similar architecture: each switch has n_a azimuths, which also represents the number of input/output ports, but compared to a hybrid switch, the store-and-forward switch buffers all incoming packets, then re-emits them FIFO. In the electronic switch packets are never lost, and all the packets undergo at first OE and then EO conversions.

The network topology under this study, that interconnects 128 servers by means of 80 identical switches with $n_a = 8$ azimuths, is presented in Fig. 3 [10], inspired from [9]. This is a 8-ary fat-tree topology [24], that represents uniquely intra-datacenter interconnects and is able to interconnect a lot of servers from the same data center by a switch of just several bidirectional ports (for example, by an optical packet switch of 8 bidirectional ports supporting 10Gbit/s [13]). Switch of level 1 (L1) represents a typical for data centers top-of-rack (ToR) switch, which interconnects 4 servers of the same rack and 4 switches of level 2 (L2). Switch L2 interconnects 4 neighbor racks and 4 switches of level 3 (L3), which, in its turn, interconnects 8 switches L2, or 8 groups of racks. One can represent the same topology in the terms of pods: there are 8 pods, interconnected by 16 L3 core-switches, in each pod there are 4 L2 aggregate layer switches and 4 L1 access layer switches interconnecting 16 servers. This topology as well offers for all server pairs the same bisection bandwidth [9] and allow load balancing, as per not-from-the-same-rack servers pair there are several equal paths possible. Each server has network interface cards of 10 Gb/s bit rate. Hybrid switch, presented in Fig. 1, is studied with a variable number of $n_e \in \{0, 2, 5, 8\}$ of the same bit rate, with $n_e = 0$ representing the all-optical switch case, and $n_e = 8$, that aims to represent a fully-buffered hybrid switch, where $n_e = n_a$, for comparison with electronic switch case. However, we must note that due to details of implementations of a simulator, which takes in account logical consequences of switching delays, it is possible that at the instant when the input of the switch is released, the input of the electronic buffer is

still occupied. This fact implies that even in fully-buffered hybrid switch there could be residual PLR under overflow load: when a packet finishes its bufferization, while liberating its switch input, another packet enters, which observes that there are no buffer inputs available, because previous packet is not yet completely buffered, thus being blocked. We also consider a case with an electronic switch, where all packets that enter the switch are buffered, and no packets are lost.

All links are bidirectional and of the same length $l_{link} \in \{10, 100\}$ m as typical link lengths for datacenters and LANs. The link plays role of device-to-device connection, i.e. server-to-switch, switch-to-server or switch-to-switch. Link is supposed to represent a non-wavelength-specific channel. Paths between servers are calculated as minimum number of hops, which offers multiple equal paths for packet transmission; that is very beneficial for OPS, allowing lowering the PLR thanks to load-balancing. This means that a packet has an equal probability to use each of the available paths (the same rule applies for buffered packets).

In our case we follow a Poissonian process of arrivals of new connection demands between all of the servers: connection demands arrive following the Poisson distribution with a given mean number of file transmission requests per second, which defines the load on a network. The performance of a network with different switches and protocols is studied under progressively increasing load.

V. Evaluation of Results

We present here the results of our study and their analysis that will lead us to the conclusion on how the TCP CCAs influence on the throughput on OPS network with all-optical, hybrid and electronic switches. All these reflections let us to conclude on which combination of TCP CCA and switch type is beneficial for which scenario.

To reduce statistical fluctuations, we repeated every simulation a hundred times with different random seeds for each TCP CCAs along with different n_e value, taking in account the case of electronic switch with reduced and conventional RTO.

The mean throughput for datacenters and LAN networks with 95% t-Student confidence intervals on every second point is represented in Fig. 4 and in Fig. 5 for the cases of $l_{link} = 10\ m$ and $l_{link} = 100\ m$ respectively. In each figure we represent four cases, one in each subfigure: *a)* TCP SAW, i.e. TCP SAWL with $p = 0$, *b)* TCP SAWL with $p = 4$, as the favorable value we found in [10], *c)* TCP SACK and *d)* TCP mSACK. We shall examine, for each protocol family, the throughput given by hybrid switches and optical switches for all link lengths; then the comparative performance of electronic switches will be studied in the second half of this section.

First of all, we can validate our results for the case with an all-optical network, i.e. $n_e = 0$, under TCP SAW and TCP mSACK (as it is a direct implementation of TCP mAIMD presented in [9]): the throughput values presented in Fig. 4a, 5a, 4d, 5d, have the same order of magnitude and follow a very similar curve for the same numbers of requests per second as those presented in [9]. The slight differences could be explained by the difference in the file size distribution implemented in the simulator, which has some arbitrary parameters, as well as by possible differences in the way of a load implementation, i.e. connection demand arrival fashion. Authors of [9] implemented the way of arrival of connections, which is specific for the MapReduce application run in a datacenter, when in our case it is a simple Poissonian process. Nevertheless, the results are close enough to validate simulation method.

For the case of the datacenter with hybrid switches paired with TCP SAW in Fig. 4a, 5a, we can see that hybrid switches already gives us slightly better performance than bufferless switches till the overflow load, starting from 10^7 file transmission demands per second, but then keeps its performance, while with bufferless switches the datacenter's throughput drops, so we gain more than 100% for $l_{link} = 10$ m, and more than 30% for $l_{link} = 100$ m at high loads. This could be explained by the fact that the SAW CCA on a server may consider a data packet that was buffered even only once as lost after RTO expiration, and retransmit by sending same data packet. However, the acknowledgment of supposedly lost but only buffered packet is received shortly after retransmission and before its RTO expiration. Then CCA proceeds by sending the next data packet. In all-optical network the next data packet will be sent only upon receiving the acknowledgment of retransmitted packet, as the packet considered to be lost will be lost indeed, thus apparently decreasing the throughput on the overflow load. Such retransmissions in a network with hybrid switches may be considered as wasteful, while in all-optical network they are a necessity.

The higher RTT induced by hybrid switches, but taken into account in TCP SAWL by design to get rid of wasteful retransmissions, lets us increase the throughput for the case of hybrid switch by up to 50% in comparison to TCP SAW for the case $l_{link} = 10$ m (Fig. 4b) and by at least 20% for $l_{link} = 100$ m (Fig.5b), at high load, i.e. more than 10^5 connection demands per second. The case of TCP SAWL for all-optical network almost doesn't influence the throughput, thus making the gain by hybrid switch paired with SAWL more important: at high load (more than 10^7 requests/s) the throughput is increased by a factor of 3 and 1.5 in the case of $l_{link} = 10$ m and $l_{link} = 100$ m respectively.

In general, for the cases of TCP SAW and TCP SAWL the number of n_e doesn't much influence the throughput. Yet, the bigger n_e, the better the throughput, but already with $n_e = 2$ the average throughput differs only by 5% at worst from the case with $n_e = 8$ under high load.

While reviewing the TCP mAIMD family of TCP CCAs, we can witness in Fig. 4 and 5 that TCP SACK and TCP mSACK paired with hybrid switches keep its high performance for two cases of $l_{link} = 10$ m and $l_{link} = 100$ m and are not sensitive to link length changes.

In comparison to TCP SAWL, as best-performing CCA from the SAW family, TCP SACK and TCP mSACK have same order of magnitude of throughput for $l_{link} = 10$ m and outperform the TCP SAWL by 40% already for $n_e = 2$ in the case of $l_{link} = 100$ m under high load. It is worth mentioning that in the all-optical network case, on the contrary, TCP SAW outperforms TCP SACK and TCP mSACK, as shown in [9].

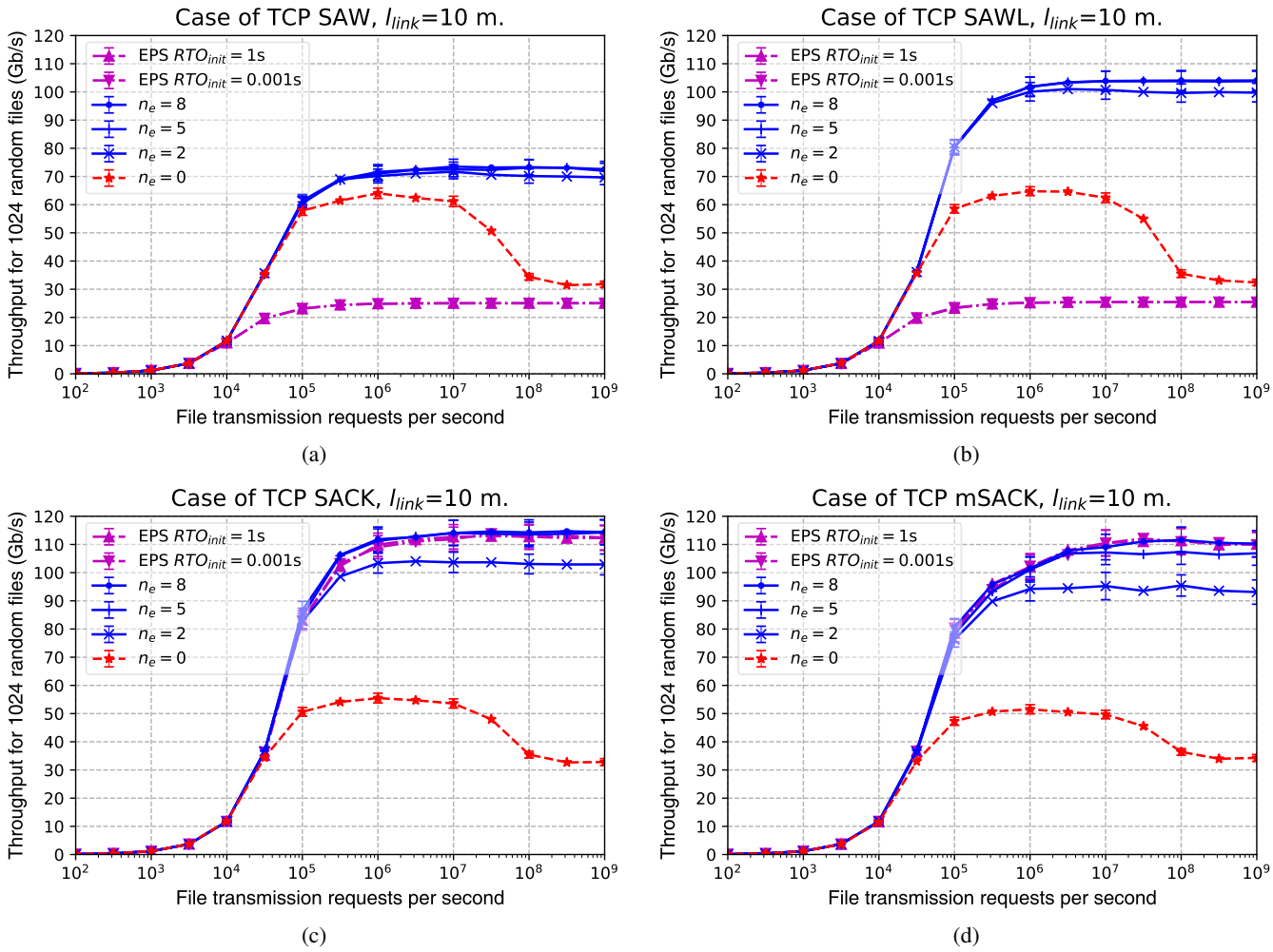

Fig. 4: Datacenter or LAN network with $l_{link} = 10$ m throughput dependence on either number of buffer I/O ports n_e or electronic switch with difference initial RTO timer for TCP: a) SAW , b) SAWL , c) SACK , d) mSACK

For the cases of TCP SACK and TCP mSACK the choice of n_e plays a more important role than in TCP SAWL. The throughput changes in average by 15% and by 17% in TCP SACK and TCP mSACK respectively under high load.

If one compares the performance of TCP SACK and TCP mSACK for the cases of datacenter with hybrid switches, it is noticeable that in general TCP SACK achieves its maximum at a lower load than TCP mSACK: for example TCP SACK achieves its maximum at 10^6 connection requests per second, when TCP mSACK achieves it only at 10^7 connection requests per second. Also TCP SACK achieves slightly better performance under high load. Yet, TCP mSACK could be regarded as worthwhile candidate for implementation in a datacenter as it could show superior performance on longer links spans and is less complex in terms of algorithmic steps, without "congestion avoidance" and "fast recovery" phases.

In general, no matter what CCA, we can witness that the hybrid switch is a robust solution for heavily loaded networks, which keeps its maximum performance and doesn't saturate as easily as solutions on all-optical switches. TCP SACK seems to be a best candidate for hybrid switches, as it keeps the best possible performance no matter what is the link length in a datacenter network.

In order to fully evaluate the performance of hybrid switches, we present the performance of networks with electronic switches, which emulates current existing EPS technology. As was said earlier before, it is important to take into account the conventional initial RTO time of 1 s [19], and not only the value of 1 ms. That said, in Fig. 4, 5 we can witness this value doesn't influence the throughput in EPS. This could be explained by low losses in EPS, incomparable neither to all-optical network nor to the cases of network with hybrid switch with $n_e < n_a$, as we simulate buffers of limitless volume. These low losses lead to the fact that retransmissions almost never occur, and each packet is sent only once.

When we review the performance of a network with electronic switches, under TCP SAW and TCP SAWL, one can witness that even all-optical network has much superior performance. This is explained by the fact that all-optical switches operate in a cut-through mode of operation, while electronic ones operate in a store-and-forward way. Thus we add to RTT a several times of duration τ, the number of which depends on how many switches packet will see along the way from source server to the destination server. Higher RTT will lead

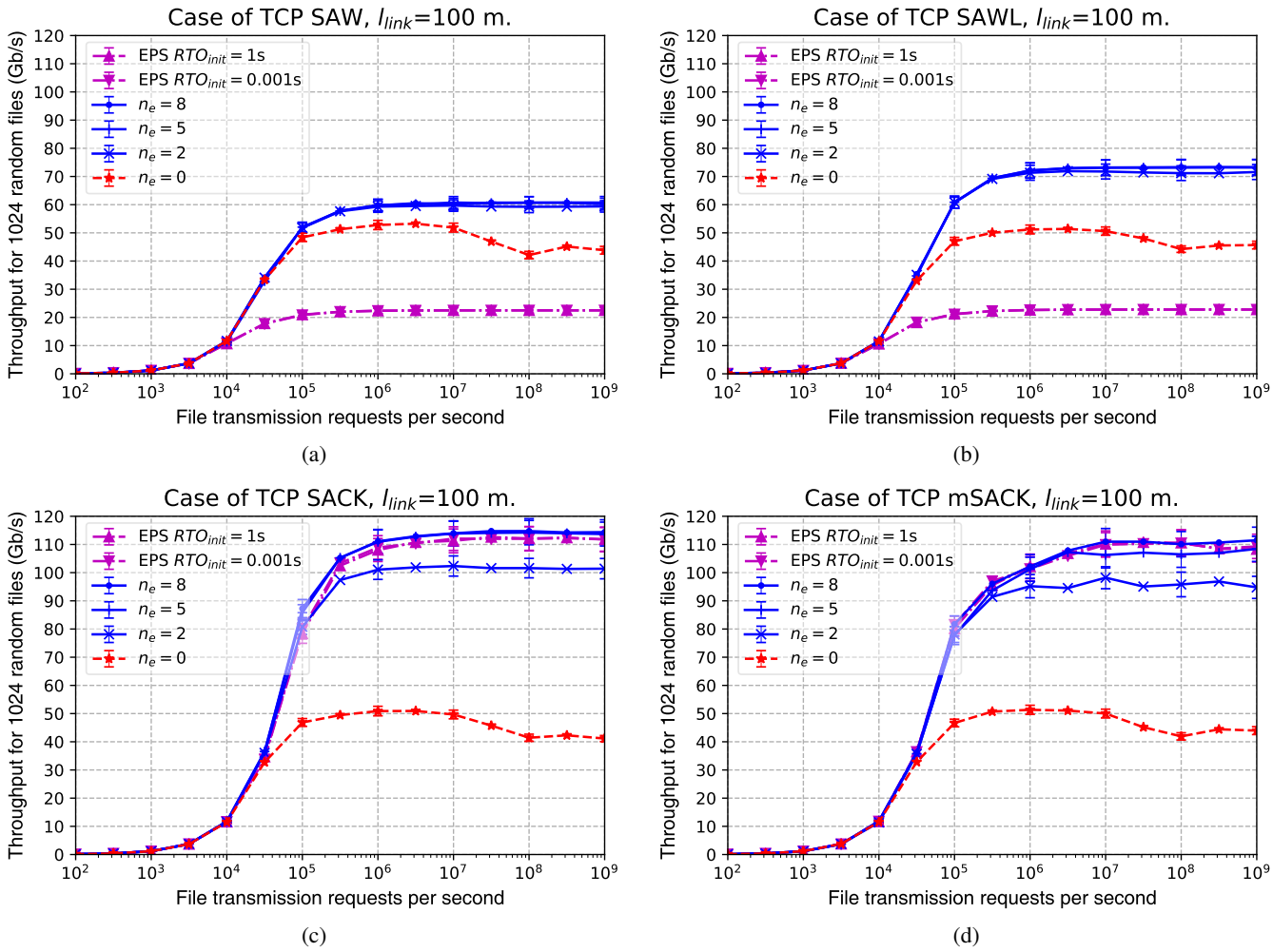

Fig. 5: Datacenter or LAN network with $l_{link} = 100$ m throughput dependence on either number of buffer I/O ports n_e or electronic switch with difference initial RTO timer for TCP: a) SAW , b) SAWL , c) SACK , d) mSACK

to poor performance when there is only one packet in flight.

However, this is not true for TCP mSACK, where the throughput of the network consisting of electronic switches achieves the same throughput as network with fully-buffered hybrid switches, i.e. when $n_e = n_a$. As for TCP SACK, the throughput of a network with electronic switch is less only by $1\% \ldots 2\%$ than throughput of the network with hybrid switches. This could be explained as well by store-and-forward mode of operation in all-electronic case, while in hybrid switch there are some packets that are switched directly to the output, i.e. experience cut-through mode, however some still stored in buffer, i.e. experience store-and-forward mode of operation. We note, that even with small number of buffer input ports, e.g. $n_e = 2$, hybrid switch let us achieve throughput, which is close to a case with electronic switch case. That fact let us conclude on possible use of hybrid switches. Additionally, even with the fully-buffered hybrid switch, that entails a reduction of OE and EO conversions: Samoud et al. in [8] already indicated that a hybrid switch OEO-converts only a small proportion of packets, which ought to give it a significant advantage in energy consumption, though we haven't yet determined how these results scale over the whole network.

Apart from the results that we presented here, we as well tested the case with conventional initial RTO time of 1 s [19] for hybrid switches under different TCP CCAs. The results of such tests had shown significant drop of the throughput under the high load for all-optical and hybrid switches (except for fully-buffered hybrid switch). Such behavior is a result of influence of all-optical part of hybrid switch and, as it was shown in [9] for all-optical switches, initial RTO of 1 s is far from optimal. This fact let us to conclude that reduction of initial RTO towards 1 ms is a crucial parameter of TCP CCA when it comes to application to network consisted of hybrid or all-optical switches.

In general it could be concluded that TCP CCAs enable the use of hybrid switches in data center networks and their specific design is a necessity while implementing new OPS solutions: it is crucial to adjust initial RTO towards 1 ms; while using SAW family of CCA, it is crucial to adjust the RTO calculus by several packet durations, so as the system wouldn't waste its resources while performing unnecessary retransmissions.

VI. Conclusions

The datacenter networks could benefit from hybrid switches that have lower energy consumption than electric ones, and a higher throughput and robustness than all-optical ones, using just a few electric ports and introducing the specially designed TCP protocols. This could be applied to any type of networks and load by changing the number of the electric ports and carefully adjusting the TCP algorithm.

In this paper we completed our previous analysis by reviewing the case of networks with electronic switches under the same conditions (including TCP CCAs) as networks with hybrid or all-optical switches. We successfully showed that, under TCP SACK, a datacenter network with fully-buffered hybrid switches gives even better performance than EPS networks; and a network with hybrid switches even with fewer buffer input ports shows already close to EPS performance.

Thus, hybrid switches enable us to have a much higher network throughput than bufferless ones in a datacenter network. Pairing TCP SACK or TCP mSACK with hybrid switches becomes a robust and beneficial solution against implementation problems of OPS. Hence, it can be inferred that the necessary condition for the interest in OPS to regain momentum [3] is found.

Our future work will be dedicated to evaluating the gain in actual energy consumption of a datacenter under hybrid switches in OPS compared to EPS. All CCAs reviewed here can potentially have different energy consumptions, specifically due to the different number of retransmissions under different protocols.

References

[1] D. J. Blumenthal, P. R. Prucnal, and J. R. Sauer, "Photonic packet switches: Architectures and experimental implementations," *Proc. IEEE*, vol. 82, no. 11, pp. 1650–1667, Nov. 1994, ISSN: 0018-9219. DOI: 10.1109/5.333744.

[2] P. Gambini, M. Renaud, C. Guillemot, F. Callegati, I. Andonovic, B. Bostica, D. Chiaroni, G. Corazza, S. L. Danielsen, P. G. and P. B. Hansen, M. Henry, C. Janz, A. Kloch, R. Krahenbuhl, C. Raffaelli, M. Schilling, A. Talneau, and L. Zucchelli, "Transparent optical packet switching: Network architecture and demonstrators in the keops project," *IEEE J. Sel. Areas Commun.*, vol. 16, no. 7, pp. 1245–1259, Sep. 1998, ISSN: 0733-8716. DOI: 10.1109/49.725193.

[3] J. R. de Almeida Amazonas, G. Santos-Boada, and J. Solé-Pareta, "Who shot optical packet switching?" In *Int. Conference on Transparent Optical Networks (ICTON)*, Jul. 2017.

[4] A. Kimsas, H. Øverby, S. Bjornstad, and V. L. Tuft, "A cross layer study of packet loss in all-optical networks," in *Proceedings of AICT/ICIW*, 2006.

[5] C. Ware, W. Samoud, P. Gravey, and M. Lourdiane, "Recent advances in optical and hybrid packet switching," in *Int. Conference on Transparent Optical Networks (ICTON)*, Trento, Italia, Jul. 2016.

[6] S. Ibrahim and R. Takahashi, "Hybrid optoelectronic router for future optical packet-switched networks," in *Optoelectronics - Advanced Device Structures*, S. L. Pyshkin and J. Ballato, Eds., Rijeka: InTech, 2017, ch. 04. DOI: 10.5772/67623.

[7] W. Samoud, C. Ware, and M. Lourdiane, "Investigation of a hybrid optical-electronic switch supporting different service classes," in *Photonics North*, vol. 9288, Montreal, Canada, May 2014, pp. 928809, 1–6.

[8] ——, "Performance analysis of a hybrid optical-electronic packet switch supporting different service classes," *IEEE J. Opt. Commun. Netw.*, vol. 7, no. 9, pp. 952–959, Sep. 2015, ISSN: 1943-0620. DOI: 10.1364/JOCN.7.000952.

[9] P. J. Argibay-Losada, G. Sahin, K. Nozhnina, and C. Qiao, "Transport-layer control to increase throughput in bufferless optical packet-switching networks," *IEEE J. Opt. Commun. Netw.*, vol. 8, no. 12, pp. 947–961, Dec. 2016.

[10] A. Minakhmetov, C. Ware, and L. Iannone, "Optical networks throughput enhancement via tcp stop-and-wait on hybrid switches," in *Optical Fiber Communication Conference*, Optical Society of America, 2018, W4I.4. DOI: 10.1364/OFC.2018.W4I.4.

[11] E. Blanton, M. Allman, L. Wang, I. Jarvinen, M. Kojo, and Y. Nishida, "A conservative loss recovery algorithm based on selective acknowledgment (SACK) for TCP," RFC Editor, RFC 6675, Aug. 2012.

[12] X. Ye, P. Mejia, Y. Yin, R. Proietti, S. J. B. Yoo, and V. Akella, "Dos - a scalable optical switch for datacenters," in *2010 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS)*, Oct. 2010, pp. 1–12.

[13] R. Takahashi, T. Nakahara, Y. Suzaki, T. Segawa, H. Ishikawa, and S. Ibrahim, "Recent progress on the hybrid optoelectronic router," in *2012 International Conference on Photonics in Switching (PS)*, Sep. 2012, pp. 1–3.

[14] Y. Yin, R. Proietti, X. Ye, C. J. Nitta, V. Akella, and S. J. B. Yoo, "Lions: An awgr-based low-latency optical switch for high-performance computing and data centers," *IEEE J. Sel. Topics Quantum Electron.*, vol. 19, no. 2, pp. 3 600 409–3 600 409, Mar. 2013, ISSN: 1077-260X. DOI: 10.1109/JSTQE.2012.2209174.

[15] T. Segawa, S. Ibrahim, T. Nakahara, Y. Muranaka, and R. Takahashi, "Low-power optical packet switching for 100-gb/s burst optical packets with a label processor and 8 x 8 optical switch," *J. Lightw. Technol.*, vol. 34, no. 8, pp. 1844–1850, Apr. 2016, ISSN: 0733-8724. DOI: 10.1109/JLT.2015.2512844.

[16] X. Yu, H. Gu, K. Wang, M. Xu, and Y. Guo, "Mpnack: An optical switching scheme enabling the buffer-less reliable transmission," vol. 10244, 2017. DOI: 10.1117/12.2264568.

[17] N. Terzenidis, M. Moralis-Pegios, G. Mourgias-Alexandris, K. Vyrsokinos, and N. Pleros, "High-port low-latency optical switch architecture with optical feed-forward buffering for 256-node disaggregated data centers," *Opt. Express*, vol. 26, no. 7, pp. 8756–8766, Apr. 2018. DOI: 10.1364/OE.26.008756.

[18] T. CHU, L. QIAO, W. TANG, D. GUO, and W. WU, "Fast, high-radix silicon photonic switches," in *Optical Fiber Communication Conference*, Optical Society of America, 2018, Th1J.4. DOI: 10.1364/OFC.2018.Th1J.4.

[19] V. Paxson, M. Allman, J. Chu, and M. Sargent, "Computing TCP's retransmission timer," RFC Editor, RFC 6298, Jun. 2011.

[20] M. Allman, V. Paxson, and E. Blanton, "TCP congestion control," RFC Editor, RFC 5681, Sep. 2009.

[21] N. Agrawal, W. Bolosky, J. Douceur, and J. Lorch, "A five-year study of file-system metadata," *ACM Trans. Storage*, vol. 3, no. 3, 2007.

[22] P. J. Argibay-Losada, K. Nozhnina, G. Sahin, and C. Qiao, "Using stop-and-wait to improve tcp throughput in fast optical switching (fos) networks over short physical distances," in *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, Apr. 2014, pp. 1312–1320. DOI: 10.1109/INFOCOM.2014.6848064.

[23] "IEEE standard for ethernet," *IEEE Std 802.3-2015 (Revision of IEEE Std 802.3-2012)*, pp. 1–4017, Mar. 2016. DOI: 10.1109/IEEESTD.2016.7428776.

[24] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," in *Proceedings of the ACM SIGCOMM 2008 Conference on Data Communication*, ser. SIGCOMM '08, Seattle, WA, USA: ACM, 2008, pp. 63–74, ISBN: 978-1-60558-175-0. DOI: 10.1145/1402958.1402967.