An Improvement Over TCP Vegas To Enhance Its Performance In Optical Burst Switching Networks

Reza Poorzare†, Anna Calveras, and Siamak Abedidarabad

† *Department of Network Engineering, Universitat Politècnica de Catalunya – UPC BarcelonaTech, Barcelona, Spain*
reza.poorzare@upc.edu
Department of Network Engineering, Universitat Politècnica de Catalunya – UPC BarcelonaTech, Barcelona, Spain
anna.calveras@upc.edu
Young Researchers and Elite Club, Ardabil Branch, Islamic Azad University, Ardabil, Iran
s.abedi@iauardabil.ac.ir

Abstract— The demand for high bandwidth on the Internet is growing drastically, and one of the solutions for tackling this problem is using optical networks. Burst switching is one of the techniques that can be used in optical networks to handle high traffic. Aside from many advantages that this technique has, it suffers from a big flaw called burst contention. Optical Burst Switching (OBS) is a switching technique without any buffering system. As a result, when two bursts are trying to reserve one resource, one of them drops. This drawback can have a significant impact on the performance of some protocols like TCP because they have not been designed to perform in a network without any queuing system and cannot distinguish a drop is because of the congestion or contention. In this paper, a new algorithm called AVGR (Average of RTTs) is proposed based on some mathematical equations to prevent the degradation of TCP. It tries to calculate averages for some RTTs in three different periods. Then base on the obtained results, the congestion control mechanism will be modified. The primary goal of the algorithm is to determine the current status of the network and make proper decisions based on it.

***Keywords*— Optical Communication, Optical Burst Switching (OBS), Data Burst Contention, Average, and Transmission Control Protocol.**

I. INTRODUCTION

Some TCP variants like Vegas cannot work appropriately in optical networks, which using OBS switching technique, because these types of protocols have not been designed to work in networks without any queuing methods. Researchers have done much studying on this issue to improve the performance of optical networks running under protocols like TCP. A large number of previous enhancements have modified TCP or OBS to adapt them to optical networks [1-11].

During the last decades, various switching techniques have been introduced for optical networks, such as optical circuit switching (OCS), optical packet switching (OPS), and optical burst switching (OBS). Among these techniques, OBS is the one that can attain an efficient and dynamic bandwidth allocation for handling a large amount of traffic on the Internet [2-3]. Control packets and data bursts are fundamental components of OBS networks [4]. Packets are aggregated into collections called data bursts in edge nodes. These nodes that are responsible for making bursts are called ingress nodes. Ingress nodes are ones that receive IP packets and assemble them in order to create data bursts. After creating a burst, it is sent through the network and received at egress nodes, and then it is disassembled to be delivered to final destinations.

Before sending a burst, the edge node launches a control packet to set up the virtual paths, which the corresponding burst will travel through. The corresponding burst is sent without receiving an acknowledgment indicating the success or failure of the path reservation, as it is called one-way reservation.

Because of the bufferless nature of the OBS technique, when a source is not available, bursts are not queued while traveling through the network. Being bufferless and deploying a one-way reservation can lead to a problem called data burst contention. When two bursts are competing on one resource (i.e., a wavelength) to reserve it, one of them will be dropped [1-4]. This packet drop can mislead protocols in detecting congestion status in the network, especially protocols such as TCP, which assumes that every packet loss is because of congestion in the network.

In traditional packet-switched networks, the mechanism of forwarding packets is different. As a result, in those networks, IP packets can be queued at each intermediate node when it is necessary. In this case, congestion increases due to a buffer overflow, so a packet drop can indicate full status in the network. However, OBS does not support buffering and queuing; thus, when a data burst contention occurs, the sending rate is declined by protocols like TCP Vegas, without considering the fact that the network may not be congested. In these networks, due to random data burst contention, packet loss does not necessarily reflect a congested condition. This situation can lead to a significant reduction in the performance of the networks [3-6].

From the TCP view, a network is congested when packet losses or packet delays are being experienced in the network. These situations can be addressed by three duplicate ACKs (Acknowledgments) or RTOs (Re-

transmission Time Out). The former one shows a packet loss, and the latter one is a sign of congestion in the network. Moreover, Data burst creation, plus other parameters such as propagation delay, can have some adverse effects on the value of round-trip time (RTT) and make the situation worse.

Another critical problem is that when a burst drops, despite other networks in which only one MSS (Maximum Segment Size) is lost per drop, a collection of packets is eliminated, and a large number of packets are lost [7]. The reason is that a burst can carry several numbers of packets.

OBS networks use a technique called WDM (Wavelength Division Multiplexing). The bandwidth of a fiber link is so high; therefore, it is divided into different channels that can work individually without interfering with other channels. In this way, we can utilize the use of high bandwidth in the network. It helps networks to handle high traffic and work smoothly. So, considering this particular feature of the network, if we do not prevent performance degradation in the network, using this technique seems obsolete [3].

To sum up, networks using the OBS technique suffer from an issue called data burst contention. It happens because the networks do not employ buffers, and when two bursts are competing on the same wavelength to reserve, one of them will be dropped. This problem can mislead some protocols like TCP in congestion detection. As a result, the sending rate cannot be adjusted appropriately, so the performance of the network is affected. Motivated by this compelling scenario, in this paper, we introduce a new algorithm called AVGR to prevent the performance reduction of these networks. This approach is based on some mathematical equations to obtain several averages in the network. Afterward, it will adjust the congestion window (cwnd) size to attain acceptable performance. Comparing the simulation results for this algorithm with those to TCP Vegas showed a considerable increase in the performance and number of delivered packets. The rest of the paper is organized as follows. Section 2 highlights the related works. Section 3 includes a preliminary discussion about TCP Vegas and OBS features as the background of the research. The AVGR algorithm will be presented in Section 4. Section 5 includes packet-level simulation results, and finally, concluding remarks are given in Section 6.

II. Works Related

A tremendous number of researches has been done to decrease the effect of data burst contention on the performance of OBS networks. Some of the recent advances are included in this section.

An algorithm called FRPI was introduced in [1] to increase the performance of the network. It is a two-step algorithm to achieve its purpose. First, by using fuzzy logic, it detects the status of the network and finds out the amount of traffic in the network, and then adjusts the sending rate. In the second step, by using some thresholds and comparing current RTTs to them, it increases, decreases, or does not change the cwnd size. This algorithm has a significant effect on the performance of the network and can improve it significantly.

A threshold-based algorithm was introduced in [2] to adapt the TCP Vegas to OBS networks. This method is an RTT-based algorithm and tries to find appropriate sizes for the cwnd. It employs a threshold in order to attain its aim. If the number of RTTs that sizes exceed the minimum RTT is more than the threshold, it indicates congestion in the network. Otherwise, the network is not full. The primary purpose of the algorithm is reducing the negative impact of contentions on the performance of TCP Vegas.

A traffic control based scheme on the contention resolution process was introduced in [3] as a quality of service (QoS) mechanism to propose service level agreement (SLA) for optical burst switching technique. Traffic control based on the contention resolution process (TCCR) was used in this paper, which tries to improve three contention resolution mechanisms, i.e., burst aggregation (BA), extra offset time, and fiber delay line (FDL).

Based on the result of the research in [4], burst contention can cause throughput degradation on the performance of Fast TCP Vegas. The first goal of this research is to investigate Fast TCP over OBS networks. Then, it analyses the behavior and stability of this protocol over the network. Finally, it shows that employing random burst contentions supported by burst retransmission can stabilize this protocol over OBS networks.

A new approach of signaling for OBS networks named inverse two-way signaling was introduced in [5]. The most critical difference between two-way signaling and inverse two-way signaling is, in two-way signaling, the collection of links' states is after the completion of burst assembly. However, in inverse two-way signaling, link-state collection and burst assembly are carried out simultaneously. Simulation results show that the end-to-end delay for inverse two-way signaling and one-way signaling are similar. However, the loss rate of inverse two-way signaling is more than two-way signaling but less than one-way signaling, so it shows a tradeoff between them.

An innovative approach in dealing with the contention problem was proposed in [6]. In order to solve the problem, a contention-free OBS ring topology was proposed. The goal is to design a multi-transceivers OBS ring scheme, in which all data wavelengths are slotted, and slots on different wavelengths are not synchronous but irregularly distributed. In this topology, each node has a wavelength for itself, and for creating control packets and slots, they perform periodically. Moreover, slots on different data wavelengths are irregularly distributed but not synchronously. This scheme is categorized under a multi-transceivers one.

An attempt to find the weakness of TCP/OBS networks was made in [7]. Results indicate that data bursts that traverse through OBS networks are not secure. The reason behind this insecurity is that before sending a

burst, its corresponding control packet is sent through an out-band link, so there is a probability that someone can manipulate or duplicate it. In this case, he can steal the data inside the corresponding burst. There are different ways for performing these kinds of attacks, such as burst header flooding attack, fake burst header attack, denial of service attack, land attack, malicious burst header injection, and circulating burst header attack.

In burst header flooding attacks, a malicious attack compromises a node, then copies a header and floods the next nodes by using these copies, and those nodes try to reserve resources for the fake bursts. The aim of this attack is that the other nodes cannot reserve resources for their data bursts because they are reserved by the fake ones. In fake burst header attacks, intruders try to inject a fake control packet into an intermediate node. As a result, they can route the data burst toward fake destinations.

In denial of service attacks, intruders try to make unoccupied wavelengths busy in intermediate nodes, so when new data bursts arrive at intermediate nodes, they cannot find free wavelengths to be routed. As a result, they are dropped.

For performing land attacks, attackers try to compromise core nodes, then make a copy of the burst control packet and change its destination to the source address. The purpose of this attack is to waste the resources of networks.

In malicious burst header injection, attackers try to introduce a fake control packet at the right time so that they can misguide the corresponding data burst.

In a circulating burst header attack, intruders try to compromise two nodes to initiate the attack. Then they create a control packet to circulate between those two nodes. The goal of this attack is causing a delay in reaching the data burst to their destinations.

A three-way process is used in [8] to utilize the available bandwidth of the link. The first step is investigating the effect of buffering within a stream-line effect framework. Then by exploiting the upsides of the stream-line effect, an optimized request provisioning is tested. Finally, simulations are done to obtain the results in case of proving the higher maximal wavelength efficiency.

The importance of the burst loss ratio was studied in [9]. This paper studies a new method that is a combination of buffering and retransmission to solve the contention problem in order to decrease the value of BLR in the network.

Because the paths for control packets and their corresponding data are separated, the network is sensitive to some threats like denial- or degradation-of-service attacks. This problem has been addressed in [10]. First, it explains the problem and the effects it can have on the functionality of the network, and then it tries to find a solution based on monitoring the network's traffic on the control and data channels.

Another kind of attack that can affect the performance of OBS networks called Burst Header Packet (BHP) flooding was investigated in [11]. This study reveals that this issue can degrade the Quality of Service (QoS) in OBS networks. It deploys a method based on Machine Learning (MA) to calculate the risk of the attack and then proposes a technique called decision tree-based architecture to solve the problem.

An investigation of blocking probability and the delay of an asynchronous single-wavelength optical buffer was done in [12]. The Level Crossing (LC) approach was used to get the integral equations for modeling the dynamics of the buffer. Then, a method was introduced to solve the equations. Finally, mathematical expressions of blocking probabilities and the delays for the optical buffer with general burst size distribution was introduced.

The Openscale architecture was proposed in [13]. This architecture tries to make some improvements in high scalability, fault tolerance, and effective load balancing in OBS networks. In order to achieve its goal, it employs the clustering coefficient and average path length to approximately evaluate the network's vicinity connectivity and arbitrary reachability, respectively.

An investigation of a horizon-based single-channel multi-class OBS node was done in [14]. This approach studies the horizon-based single-channel multi-class OBS networks along with the general burst size distribution. By using a machine learning technique, the per-class blocking probabilities were available.

The next section incorporates fundamental backgrounds of the research, including TCP Vegas and OBS technique explanation.

III. Preliminaries

This section aims to provide the background for the new proposed algorithm. First of all, the congestion avoidance phase of TCP Vegas needs to be explained Because TCP Vegas is the default used algorithm in this paper, and the new algorithm is created by changing the congestion avoidance phase of it. Moreover, the obtained simulation results for the proposed algorithm have been compared to the ones from Vegas. Secondly, a concise explanation is necessary for the OBS technique in order to motivate the research.

A. TCP Vegas

TCP Vegas is one of the well-known variants of TCP protocol. It measures the RTTs of a network to have a better understanding of the traffic load, and unlike other TCP variants such as NewReno, it does not wait for congestion indicators to happen in the network and tries to avoid packet losses before happening [1-3]. It calculates two throughputs and deploys them to achieve its goal. These throughputs are called Expected and Actual. Equation (1) shows how they are calculated:

$$Expected = \frac{cwnd}{BaseRTT}$$

$$Actual = \frac{cwnd}{RTT} \quad (1)$$

BaseRTT is calculated by using propagation delay and queuing time, and cwnd is the congestion window size, which specifies the number of packets that a sender can transmit without waiting for the corresponding ACKs (i.e., packets in-flight). For adjusting the next cwnd, TCP Vegas calculates a variable called Diff by using equation (2):

$$Diff = Expected - Actual = cwnd(1 - \frac{BaseRtt}{RTT}) \quad (2)$$

Then TCP Vegas deploys two thresholds denoted as α and β to adjust the next cwnd. Equation (3) shows this process:

$$cwnd = \begin{cases} cwnd + 1 & diff < \alpha \\ cwnd & \alpha \le diff \le \beta \\ cwnd - 1 & diff > \beta \end{cases} \quad (3)$$

By looking at this equation, it can be deduced that when the network is congested, the next size of cwnd is reduced by TCP Vegas to stabilize the network. When the flow of the network is reasonable, the current rate is kept by TCP Vegas, and the sending rate is not changed. Otherwise, when the network is empty, the next size of cwnd is increased to utilize the use of the available bandwidth [2].

Although this approach can work fine in many networks, it is not compatible with OBS networks and is the source of the drastic reduction in the performance of these networks. Sometimes the network is not congested, and a data burst contention happens in the network. In this case, TCP Vegas assumes that the network is going to be congested, so it starts to decrease the sending rate despite the fact that it is not necessary. Moreover, some TCPs like Vegas are not suitably designed to work in high-speed networks. In order to utilize the high bandwidth of optical networks, these behaviors, which reduce the performance of TCP, need to be addressed to prevent performance reduction over OBS networks.

B. Optical Burst Switching

Optical Burst Switching is one of the optical switching techniques which strives to perform switching in a dynamic way. The goal of this technique is to compromise between optical packet switching (OPS) and optical circuit switching (OCS). The key difference between OBS and other switching techniques is that it sends its control packets through a reserved channel prior to sending the main data, and they try to reserve a light path for the following data. This technique of reservation is called a delayed reservation.

The main goal of OBS is to create an optical-electrical network. The electrical part is the control packets that are processed electrically through the network, and the optical part is sending the main data through the optical path, i.e., optical wavelength. Although OBS provides flexibility for the network, it needs fast switching and control technology. OBS indeed provides more efficient bandwidth utilization, but it suffers from a severe drawback, which is called data burst contention. When more than one data compete to reserve a resource (i.e., a wavelength), one of them is dropped. This flaw can degrade the performance of TCP, which assumes every packet drop is because of congestion in the network [1-3].

In this paper, we propose a novel algorithm that can distinguish packet losses by congestion from one due to contention by knowing the current status of the network. This approach can improve the performance of OBS networks when deployed. Furthermore, the proposed algorithm is able to ramp up to the high potential of optical networks when the network is empty or reduces the sending rate quickly when it is heavily congested.

IV. Proposed Algorithm to Enhance the Performance of Optical Burst Switching Networks

Because of the bufferless nature of OBS networks, some protocols such as TCP are not able to work properly. The reason behind this issue is due to the differences between exploited mechanisms in optical networks compared to copper-wired networks. In contrast to conventional networks, OBS networks are bufferless, and there is no queuing technique used within the networks. As a result, it experiences significant reduction caused by an event called data burst contention that makes some packets losses in the smooth traffic. The most severe flaw of Vegas, which is a delay-based TCP, and other loss-based TCPs such as NewReno and CUBIC, when used in OBS networks, is that they cannot differentiate packet drops caused by congestion from the ones caused by contention, and they do not have any clear strategy to distinguish these two different types of packet losses in the network which leads to confusion in the functionality of TCP. As a result, it can not perform efficiently in OBS networks and causes performance degradation. In addition to misunderstanding in the status of the network, These TCPs have not been designed to work in high-speed networks.

In this section, we intend to propose a new algorithm that can improve the performance of Vegas over OBS networks. The scenario behind this algorithm is using mathematical equations to obtain some averages, then comparing them together and set the cwnd size.

As we know, TCP keeps the value of every RTT, so if we can use this parameter to create a new algorithm, it reduces the overhead of the network. For attaining our purpose, we measure several RTTs in the network in different periods, then calculate averages for these RTTs and compare them together to set the cwnd size. AVGR

algorithm relies on some mathematical equations to get the status of the network, and based on that, adjusts the sending rate. As a result, this algorithm includes two phases. In the first phase, the calculation of the averages is done, and in the second one, decisions making based on the results of the first phase is performed.

The first step of this procedure is measuring the value of RTTs one after another. For determining the calculation ending point, a threshold is used, and when this threshold is reached, the measuring procedure is stopped, and the average of them is calculated. After that, in the second step, the next round of RTTs is measured, and their average is calculated and stored. We repeat this process for the third round, too. So, we measure the RTTs in three different periods and keep their averages.

Figure **1** shows how the first phase of the algorithm works. In the first phase, the algorithm tends to calculate the averages for the particular number of RTTs. This process will be repeated three times to have a better understanding of the network's status. Because by having a clear view of the network, the accuracy of the decisions gets higher.

When the first phase is over, the second phase initiates. In this phase, the algorithm deploys some equations to set the size of the congestion window. All of these equations are trying to look at the current status of the network, and then based on the load of the traffic, try to adjust the sending rate. There is a scenario behind each equation, which will be explained.

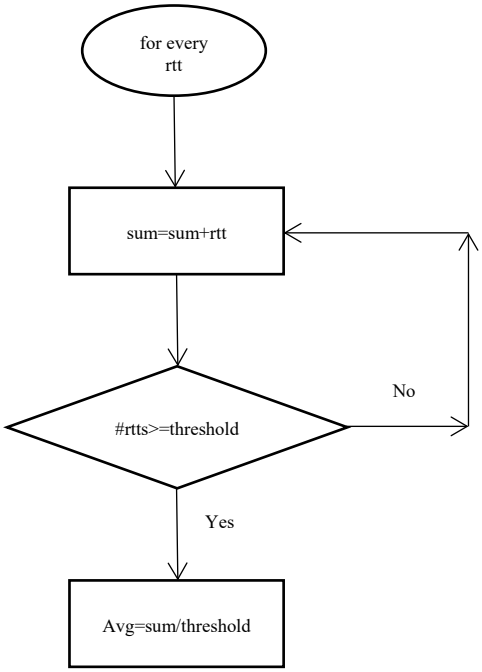

Figure 1: The first phase of the AVGR algorithm

The second phase is the brain of the algorithm. At this stage, the algorithm tries to adjust the behavior of the network by using five different equations. In this phase, the algorithm divides the network into five different working areas, and based on the current one, decides the proper reaction to be made. Figure **2** shows how decisions are made in different circumstances.

When the third average is larger than the first one, and the second one is smaller than the first one, it indicates heavy congestion in the network. It means that the RTTs experienced drastic increments, which can be signs of heavy traffic in the network. In this case, the size of cwnd will be decreased by 2 to mitigate the existing heavy load in the network over fast paces.

When the third average is larger than the second one and the second one is larger than the first one, RTTs are experiencing steady increments, so the network is getting congested gradually, but it is not full. In this case, the size of cwnd will be reduced by 1 to prevent the network from being congested heavily and forcing it to function in a normal way.

When two or more than two average values are close, it shows that the network is working smoothly. It means that neither the network is congested nor empty. If so, the sending rate is not modified in order to maintain the current normal status in the network.

```
begin
    For every rtt
        If ((avg3>avg1) && (avg2<avg1)) then
            //The network is heavily congested
              cwnd=cwnd-2
        end
        If ((avg3>avg2) && (avg2>avg1)) then
            //The network is congested
              cwnd=cwnd-1
        end
        If ((avg3=avg2) || (avg3=avg1) || (avg2=avg1)) then
            //The network is normal
              cwnd=cwnd
        end
        If ((avg3<avg2) && (avg2<avg1)) then
            //The network is partially empty
              cwnd=cwnd+1
        end
        If ((avg3<avg2) && (avg2>avg1)) then
        end
    end
end
```

Figure 2: How AVGR adjusts the congestion window size in different situations

When the third average is smaller than the second one and the second one is smaller than the first one, the network is about to be empty, and if it works with the current condition, the available bandwidth will not be utilized. In such a situation, RTTs are declining steadily, and we need to increase the size of cwnd by 1 to utilize the use of the available bandwidth in the network. The key indicator of this condition is steady decrements in the value of the RTTs.

When the third average is smaller than the second one, and the second one is larger than the first one, it shows that the RTTs experienced severe decrements, and the network is empty. Therefore the available bandwidth is not used efficiently, and we need to increase the size of the cwnd by 2. This increasing aims to prevent underutilization of the available bandwidth and ramps up to the full potential of the network.

Figure **3** shows the flowchart of the second phase and the way that the sending rate is controlled.

The figure shows that the adjustment of the cwnd size is based on the current status of the network. When the network is heavily congested, it is reduced by two in order to reduce the load of the traffic in the network. When the network is congested but not massively in order to make the load of the traffic normal, the cwnd value is reduced by one. When the network is normal, the algorithm does not change anything because it is operating appropriately and does not need any manipulations. When the network is partially empty, in order to utilize the use of available bandwidth, the size of cwnd is increased by one. Finally, when the network is empty, in this case, most of the bandwidth in the network is wasted, so for preventing underutilization, the size of cwnd is increased by two.

Using this algorithm helps the protocol to detect the exact status of the network and adjusts the sending rate accurately. In this way, the effect of the contention is eliminated because the sending rate is not adjusted based on the packet loss, it is adjusted by looking at the current condition of the network.

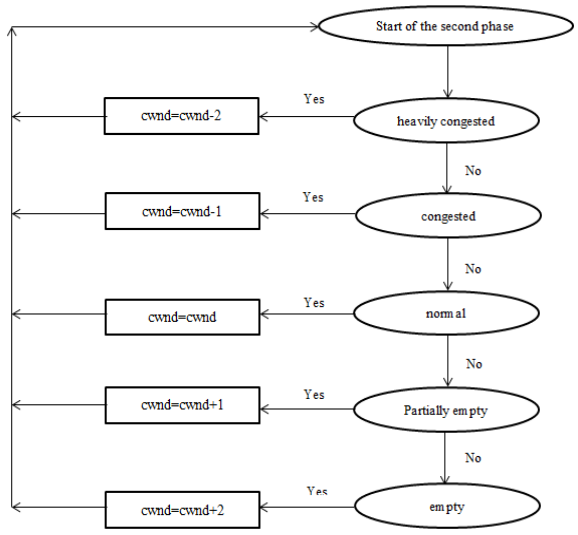

Figure 3: How the second phase of the algorithm works

V. Packet-level Simulation Results

In order to create the AVGR algorithm, we had to make some modifications to TCP Vegas. For achieving this purpose, we changed the congestion avoidance phase of TCP Vegas completely. As a result, instead of a loss-based TCP, we have a new delay-based one. For simulating the functionality of this new protocol, we have used the NS-2 (network simulator) to analyze our algorithm. NS-2 is a powerful simulating software implemented in Linux. It is a discrete event simulator that is suitable for researchers in order to test their new designs and protocols [15-16]. We had to add the OBS-ns patch and make some changes in NS-2 so it could simulate optical networks because, by default, it has some difficulties in emulating optical medium. The topology shown in Figure 4 was used as the simulating network.

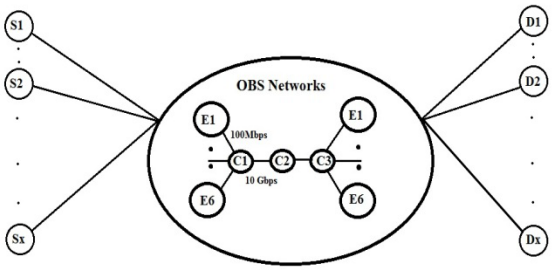

Figure 4. The topology used for the simulation

This topology has 16 edge nodes and three core nodes. An edge node is connected to a core node with a 1ms propagation delay. There are 100 wavelength channels in the links for transferring data, and the data rate of each channel is 1Mbps, so it means each edge node is connected to a core node with a 100Mbps link. Each link has eight wavelength channels for transferring control packets.

The link between core nodes has 100 wavelength channels, and the data rate of each channel is 100Mbps. Thus, core nodes are connected to each other with 10Gbps bandwidth. The links also have eight wavelength channels for transferring control packets. A mixed time/length based algorithm is used in creating data bursts in ingress nodes. The burst timeout is 5ms, and the maximum burst size is 50 Kb. The control header processing time is set to 20μs, and the offset time is 10μs. The contention probability varies in the range of $[10^{-5}, 10^{-2}]$ in order to emulate an OBS network. Finally, the simulation time is set to 10000 seconds.

A. Using Different Thresholds For Calculating The Averages

As it was mentioned, the algorithm measures different RTTs in various rounds to calculate the averages. The number of RTTs that are measured in each round called N. This number can have a significant role in the performance of the algorithm. If we choose it improper-

ly, it will affect the algorithm dramatically. Therefore the first step is to find the size for N. The main procedure of finding N is manually through extensive simulations. We have tested more than a hundred ones, and the result was impressive. It is noteworthy to say that this number can be different based on the network parameters, network topology, and the deployed protocol.

First, we show the results in Figure **5**, and then we will discuss the intriguing obtained results.

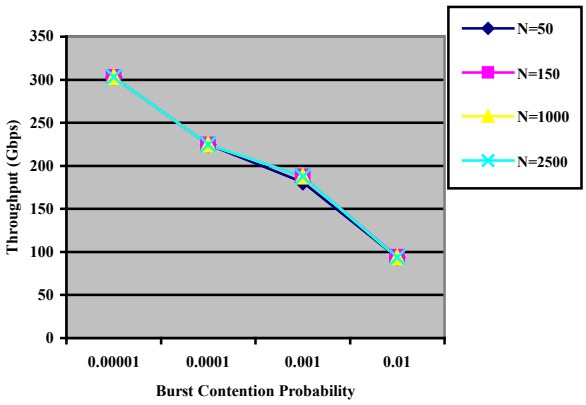

Figure 5: The obtained throughputs for different tested thresholds

Obtaining these close results convinced us to do tremendous simulations to find a relationship between numbers. The procedure continued by testing numerous thresholds in order to find a border between the values. Using the AVGR algorithm leads to some numbers that the difference between them is not significant, especially when the contention probability is high. The reason behind these results is that this algorithm provides a better view of the network and can perform with a slight number of errors compared to the conventional loss-based and delay-based TCPs.

The next step is finding the number that acts as the border between different thresholds. As a result, we continued to simulate under different thresholds, and after a large number of simulations, it was concluded that the interesting number is 137. It means that the results for the numbers less than this number are about the same (not identical but almost the same), and the results for the numbers above this line are about the same too.

By having a close look at the numbers and investigating them in detail, N=150 was chosen as the best one for the algorithm. As a result, the step by step procedure for the algorithm is:

It measures the first 150 RTTs.
It calculates the average for the first 150 RTTs.
It measures the second 150 RTTs, which means from 150 to 300.
It calculates the average for the second 150 RTTs.
It measures the third 150 RTTs, which means from 300 to 450.
It calculates the average for the third 150 RTTs.
It begins the second phase of the algorithm, and by using equations 4-8, it estimates the status of the network.
In the final step, based on the current status of the network, it adjusts the size of the congestion window.

For proving this fact that the AVGR algorithm could improve the performance of OBS networks, we compare it with one of the well-known variants of TCP called TCP Vegas. Figure **6** shows a comparison of throughputs between these two.

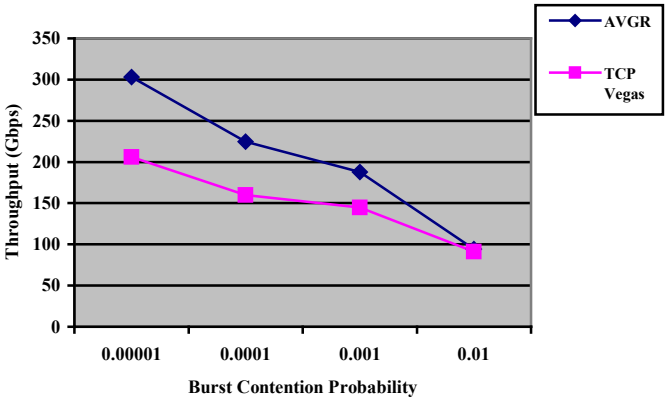

Figure 6: Throughput comparison between AVGR and TCP Vegas

This figure indicates that AVGR outperforms TCP Vegas in terms of throughput as the contention probability rises in the network. Moreover, our topology has similarities to [2], and comparing the throughputs of two algorithms reveals that AVGR can attain better performance. Plus, comparing the throughput with [4] also shows better performance for the AVGR algorithm in terms of throughput. For having a detailed analysis, we can compare the results with [1] and [17], as the three pieces of research have the same topology. The obtained results for the AVGR is almost 50% better than [17]. However, the results in [1] and the AVGR are almost the same.

The reason for the superiority of the AVGR is the better view of AVGR from the network.

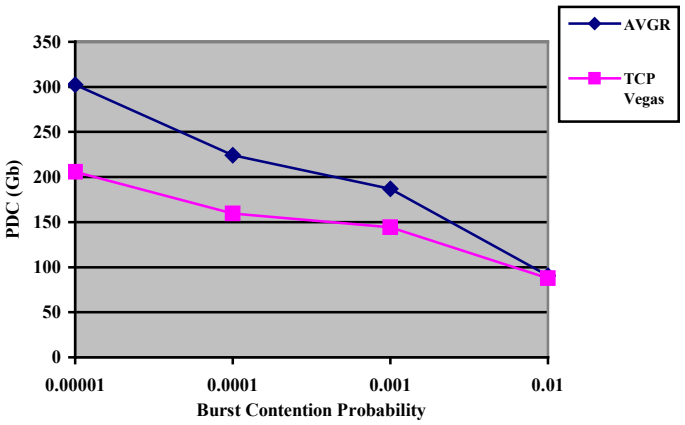

Figure 7: PDC for AVGR and TCP Vegas

When a burst drop happens, it can find out the source of it by knowing the current condition of the network. The throughput is one of the most important KPIs (Key Performance Indicators) for a network, and our new algorithm can show a significant enhancement for this KPI.

The next parameter to be compared is the packet delivery count (PDC).

The number of delivered packets in a network can be another KPI in order to show its functionality. As a result, we decided to have a comparison between our algorithm and TCP Vegas in terms of delivered packets.

Figure **7** shows the difference between these two methods in delivering the packets. As it was expected, AVGR can deliver more packets than TCP Vegas because of the way it approached the problem. Dividing the network into five areas of congestion helps our protocol to make the decisions properly. This efficient decision-making strategy enhances the throughput and PDC parameters when AVGR is deployed in the network.

The next parameter that is necessary to be compared between these two methods is the packet Delivery ratio (PDR). This parameter can be an indicator of how well a protocol works. It shows the ratio between delivered packets and sent packets, so how a protocol is successful in delivering packets can be measured by PDR. Figure **8** shows a comparison of this parameter.

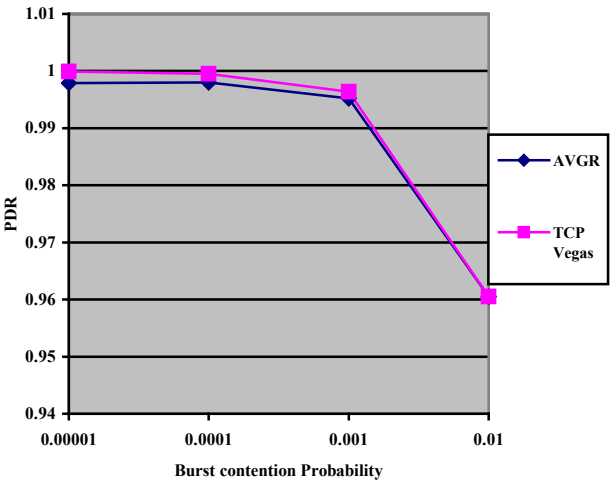

Figure 8: PDR comparison between AVGR and TCP Vegas

This figure does not show a significant difference between AVGR and TCP Vegas in terms of PDR, and considering the other parameters, this one is acceptable too for the new proposed algorithm.

In the next section, we are going to investigate the congestion window (cwnd) size for both approaches in order to find out how AVGR has this superiority over TCP Vegas.

B. A Close Look at the Congestion Window Size

Congestion windows size can play a major role in how well a protocol handles different situations in a network. It is like the heart of TCP and needs to beat in a regular and flawless way. Any false fluctuations or changes in the congestion window size can deteriorate the performance of the protocol.

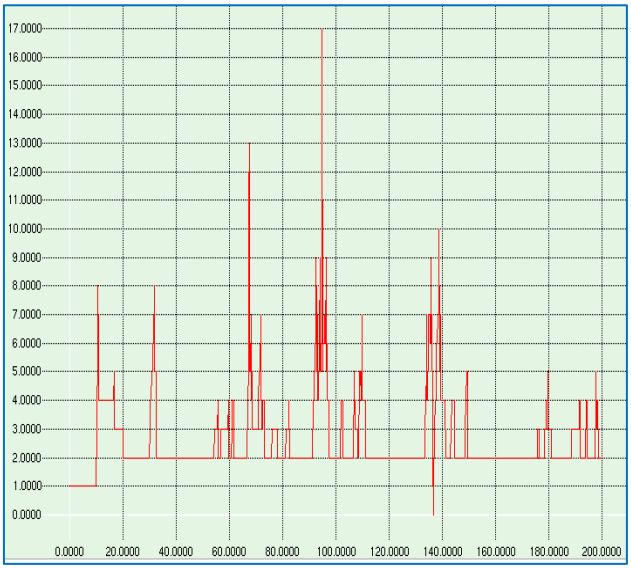

Figure 9: How AVGR controls the cwnd size when the burst contention probability is 10-2

As a result, it is essential to investigate the cwnd size in detail in order to have a closer look on the ways that the AVGR algorithm and TCP Vegas approach the problem. The reason for this is that we want to discover why AVGR functions better than TCP Vegas, and what the reason is behind the better performance of AVGR compared to TCP Vegas.

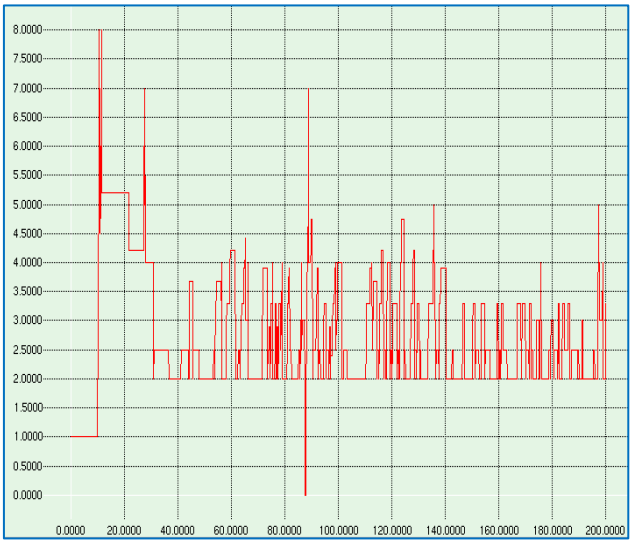

Figure 10. How TCP Vegas controls the cwnd size when the burst contention probability is 10-2

For attaining this purpose, we examine the size of cwnd in different data burst contention probabilities during 200 seconds of simulations to see the reflection of the network condition on the size of the congestion windows and to find out how these protocols react to various situations. Figure **9** and Figure **10** show cwnd sizes for AVGR and Vegas when the contention probability is 10^{-2}.

Looking at these two figures gives us some useful tips. First of all, it reveals in some periods that the network is not congested, and a burst contention happens, TCP Vegas reduces the cwnd size. However, AVGR can determine the proper status of the network and react efficiently.

Secondly, in TCP Vegas, burst contentions act as congestion indicators, so they prevent the protocol from reaching larger cwnd sizes. As a result, the performance is degraded by this misunderstanding. In contrast, this false congestion detection does not exist in the AVGR algorithm, which is one of the fundamental features of the algorithm in reaching higher throughput. When AVGR is deployed in the networks, it can attain higher sending rates which are suitable for optical networks. AVGR owes this improvement in its functionality to its flawless approach in handling the various conditions. However, it is evident from the figures that Vegas, in its best case, can reach half of the AVGR sending rate, which is a downside for this protocol when used in high-speed networks. Moreover, because of the high burst contention probability here, TCP Vegas cannot function properly, and there are many fluctuations in figure 10. In contrast, AVGR works appropriately, and in some cases, despite a burst contention occurrence, it does not reduce cwnd size wrongly.

Higher sending data rates, less number of fluctuations, having a better view of the network's condition, reaching to higher sending rates rapidly, the higher average for cwnd sizes, and persistent functionality are the upsides of the AVGR algorithm over Vegas.

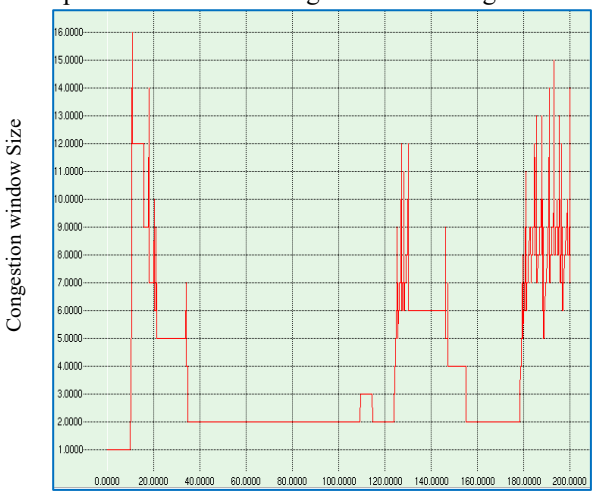

Figure 11. How AVGR controls the cwnd size when the burst contention probability is 10-3

Figure **11** and Figure **12** show cwnd sizes for AVGR and Vegas when the data burst contention is 10^{-3}. In this case, the contention probability is reduced compared to the previous scenario.

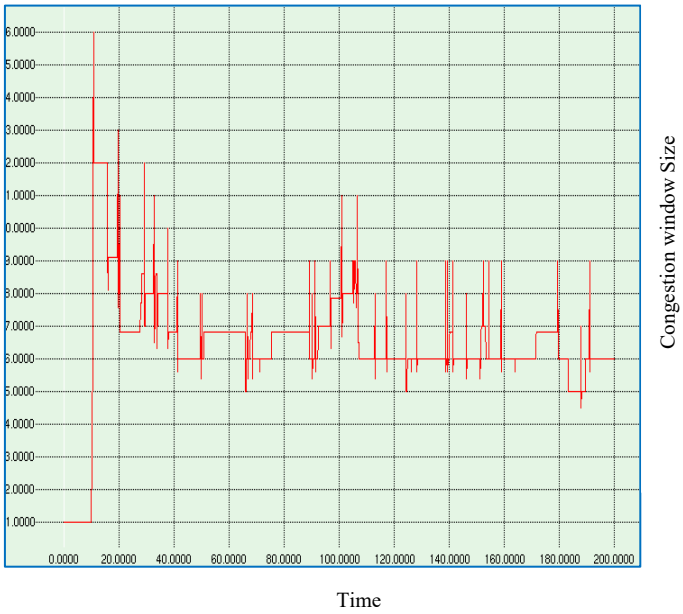

Figure 12. How TCP Vegas Controls the cwnd size when the burst contention probability is 10-3

The most important key point of comparing these two figures is that when burst contention probability is reduced, TCP Vegas can improve its functionality. However, the fluctuations remain, and the cwnd size is also small. Like 10^{-2}, AVGR has a better understanding of the network's status and can control the sending rate more appropriately. The main point of this performance is preventing cwnd size reduction when a contention happens in the network in a non-congested situation. As can be seen, our algorithm can attain sending rates that Vegas can reach rarely.

Figure **13** and Figure **14**show how AVGR and Vegas control cwnd size when the burst contention is 10^{-4}.

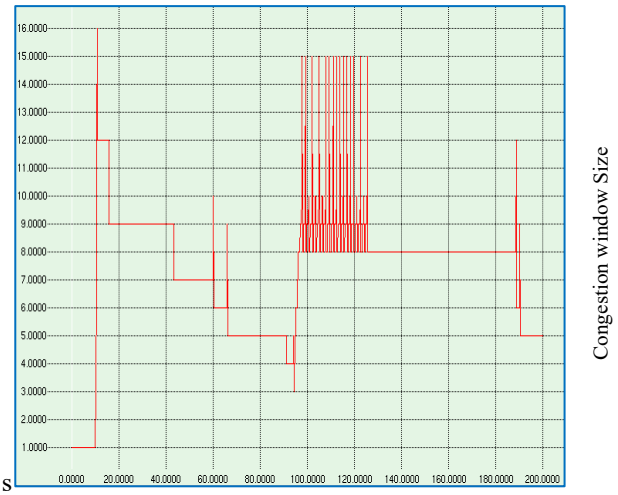

Figure 13. How AVGR controls the cwnd size when the burst contention probability is 10-4

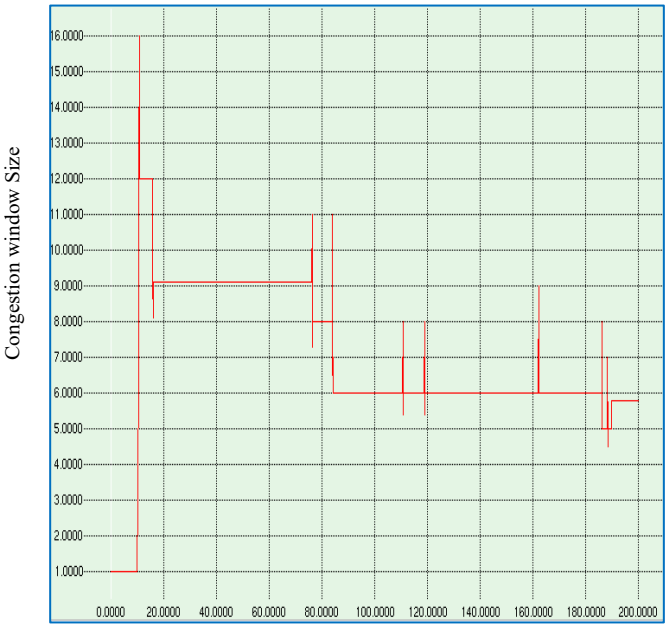

Figure 14. How TCP Vegas controls the cwnd size when the burst contention probability is 10-4

When the burst contention is 10^{-4}, because of the low value of it, TCP Vegas has a similar functionality compared to AVGR. However, they are not the same, and AVGR has slightly better performance. When the contention probability is low, it indicates that they are fewer bursts that are competing in reserving resources, and the number of contentions will be low. As a result, the misunderstanding of the congestion status of the network for TCP Vegas is reduced.

Figure **15** and Figure **16** show how AVGR and Vegas control cwnd size when the burst contention is 10^{-5}.

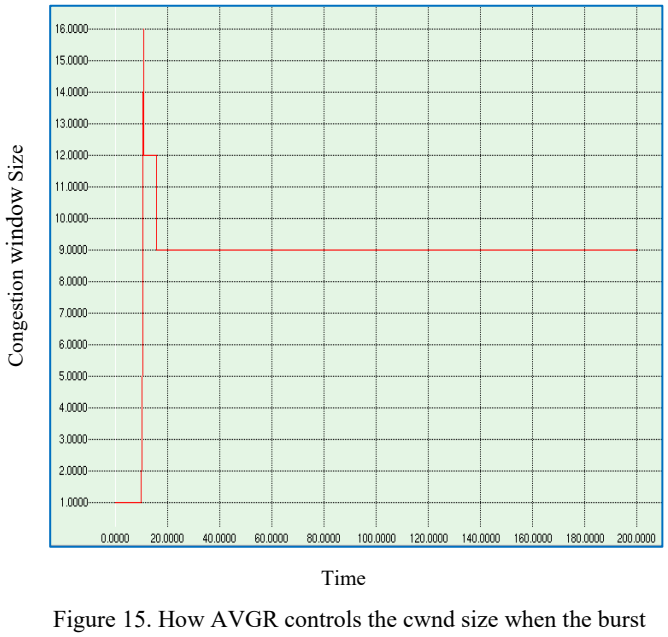

Figure 15. How AVGR controls the cwnd size when the burst contention probability is 10-5

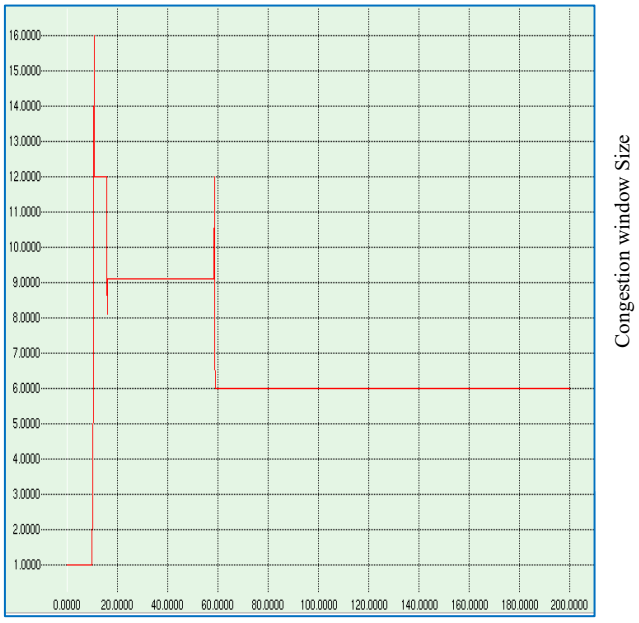

Figure 16. How TCP Vegas controls the cwnd size when the burst contention probability is 10-5

By looking at the figures, we can see that, as the contention probability is reduced, the cwnd size for both of the protocols gets closer till they are almost the same at 10^{-5}, because this number is close to zero, and it is like a situation that we do not have any contentions, and TCP Vegas function gets its standard functionality until it normally works at 10^{-5}.

By looking at the cwnd size figures, it is evident that AVGR has a better understanding of the network's status and can distinguish burst drops caused by contentions from burst drops caused by congestion. It means when a burst contention occurs, the behavior of the network is normal and can respond to a burst loss appropriately. As a result, the significant effect of false congestion detection on the performance of the network is eliminated. Moreover, AVGR can achieve higher sending rates compared to Vegas. Besides higher sending rates, the proposed algorithm fluctuates less than TCP Vegas and can have a larger average size of the congestion window.

VI. Conclusion

OBS networks suffer from a flaw called data burst contention. This drawback affects the performance of some protocols like TCP Vegas and reduces their performances significantly. In this paper, we proposed a new algorithm called AVGR based on some mathematical equations to solve the problem and improve the performance of the networks. The simulation results show that AVGR outperforms TCP Vegas in terms of throughput and PDC and has approximately the same PDR as TCP Vegas. Furthermore, the average sending

rate for the proposed algorithm is higher than the one for TCP Vegas, and its fluctuation is much less than Vegas. To sum up, higher sending rates, fewer fluctuations in congestion window size, having a better view of the network's condition, reaching a higher sending rate rapidly, a higher average value for congestion windows size, and persistent functionality are AVGR superiorities over TCP Vegas. For future works, this algorithm can be implemented in other TCP variants such as CUBIC, HighSpeed, and BBR to see that it can improve their performances or not.

ACKNOWLEDGMENT

This work has been funded by the Secretaria d'Universitats i Recerca del Departament d'Empresa i Coneixement de la Generalitat de Catalunya (2017 SGR 376) and the Spanish Government under Project PID2019-106808RA-I00 AEI/FEDER UE.

References

[1] Reza Poorzare and Siamak Abedidarabad. Improving performance of optical networks by using FRPI algorithm, Journal of Optical Communications, Nov. 2018. DOI: https://doi.org/10.1016/j.osn.2010.04.002.

[2] Shihada B, Zhang Q, Ho P-H, Jue JP. A novel implementation of TCP Vegas for Optical Burst Switched Networks, Optical Switching and Networking, Vol. 7, Issue 3, Page: 114-126, July 2010. DOI: https://doi.org/10.1016/j.osn.2010.04.002.

[3] Ekularn Dhavarudha, Chalie, Charoenlarpnopparut, and Suwan Runggeratigul. Traffic Control Based on Contention Resolution in Optical Burst, International Journal of Computers Communications and Control, Vol. 10, No. 1, Page: 49-61, Feb. 2015. DOI: 10.15837/ijccc.2015.1.461.

[4] Basem Shihada, SamiEl-Ferik, and Pin-HanHo. FAST TCP over optical burst switched networks: Modeling and stability analysis, Optical Switching and Networking, Vol. 10, No. 2, Page: 107-118, April 2013. DOI: https://doi.org/10.1016/j.osn.2012.09.001.

[5] JunlingYuan, XianweiZhou, JianpingWang, YongqiHe, and KeWang, Inverse two-way signaling scheme for optical burst switched networks, Optical Switching and Networking, Vol.9, Issue. 3, Page: 214-223, July 2012. DOI: 10.1016/j.osn.2012.02.002.

[6] JunlingYuan, XianweiZhou, JianpingWang, XuhongLi, and FuhongLin, An irregularly slotted ring scheme for contention-free optical burst switching, Optical Switching and Networking, Vol.12, Issue. 3, Page: 45-55, April 2014. DOI: 10.1016/j.osn.2014.01.002.

[7] N. Sreenath, K. Muthuraj, and G. Vinoth Kuzhandaivelu, Threats and Vulnerabilities on TCP/OBS Networks, 2012 International Conference on Computer Communication and Informatics (ICCCI -2012), Jan. 2012. DOI: 10.1109/ICCCI.2012.6158832.

[8] M.Kozak, B.Jaumard, and L.Bohaca, On the efficiency of stream line effect for contention avoidance in optical burst switching networks, Optical Switching and Networking, Vol.18, Part. 1, Page: 35-50, Nov. 2015. DOI: https://doi.org/10.1016/j.osn.2015.03.002.

[9] D.Veera Vanitha, D.Sumitha, and M.Sabrigiriraj, Analysis of Combined Buffering and Retransmission with Maintenance Activity in OBS Networks, 2018 International Conference on Current Trends towards Converging Technologies, 2018. DOI: 10.1109/ICCTCT.2018.8550882.

[10] Sudarshan S. Chawathe, Analysis of Burst Header Packets in Optical Burst Switching networks, 2018 IEEE 17th International Symposium on Network Computing and Applications, 2018. DOI: 10.1109/NCA.2018.8548071.

[11] AdelRajab, Chin-Tser Huang, and Mohammed Al-Shargabi, Decision Tree Rule Learning Approach to Counter Burst Header Packet Flooding Attack in Optical Burst Switching network, Optical Switching and Networking, Vol. 29, Pages 15-2, July 2018. DOI: https://doi.org/10.1016/j.osn.2018.03.001.

[12] Shengda Tang and Liansheng Tan, Single-wavelength optical buffers with general burst size distribution: Blocking probability and mean delay, Optical Switching and Networking, Vol. 27, Pages 1-6, Jan. 2018. DOI: https://doi.org/10.1016/j.osn.2017.05.002.

[13] Dongxu Zhang, et al., Analysis and experimental demonstration of an optical switching enabled scalable data center network architecture, Optical Switching and Networking, Vol. 23, Pages 205-214, Jan. 2017. DOI: https://doi.org/10.1016/j.osn.2016.04.002.

[14] Shengda Tang and Liansheng Tan, Analysis of Blocking Probability of Multi-Class OBS with General Burst Size Distribution, IEEE communications letters, Vol. 20, No. 11, Pages: 2153-2156, Nov. 2016. DOI: 10.1109/LCOMM.2016.2596280.

[15] Ns-2, Network Simulator, www.isi.edu.

[16] https://www.linuxjournal.com/article/5929.

[17] Siamak Abedidarabad and Reza Poorzare. Improving Performance of Optical Networks by a Probable Approach, Journal of Optical Communications, Jan. 2019. DOI: https://doi.org/10.1515/joc-2018-0221.