Journal of Theoretical and Applied Information Technology
10th September 2014. Vol. 67 No.1
© 2005 - 2014 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

A NOVEL FUZZY-BASED ADAPTIVE TIMER BURST ASSEMBLY ALGORITHM FOR OPTICAL BURST SWITCHING NETWORKS

[1]ABUBAKAR MUHAMMAD UMARU, [2]MUHAMMAD SHAFIE ABD LATIFF, and [3]YAHAYA COULIBALY
[123]Faculty of Computing, Universiti Teknologi Malaysia, 81310, Johor Bahru, Johor, Malaysia.
E-mail: [1]muabubakar2@live.utm.my, [2]shafie@utm.my, [3]coulibaly@utm.my

ABSTRACT

Optical Burst Switching (OBS) is envisioned as the intermediate candidate technology for the next generation all-optical switching networks. OBS has the capability to statistically multiplex data at the burst level and at the same time function effectively without the need for optical memory. OBS harnesses the Wavelength Division Multiplexing (WDM) capability of the optical fiber to transport large volume of data by aggregating different client packets into a larger packet known as a burst. Burst assembly that is the first process in OBS is the focus of this paper, and this is because of the crucial role burst assembly plays in the performance of a network. The performance of burst assembly affects the congestion and contention levels of the network which in turn affects the overall network performance in terms of delay and loss. In this paper, a new burst assembly algorithm that uses fuzzy logic to adjust the threshold of the timer-based burst assembly algorithm is proposed. The goal of the algorithm is to minimize loss and end-to-end delay. Furthermore, two sets of fuzzy rules have been used to investigate the effects of different fuzzy rules on the performance of the network. Through simulation, the new algorithm shows performance improvement over the conventional timer-based assembly algorithm.

Keywords: *Burst Assembly, Fuzzy Logic, Delay, Packet Loss Ratio, Optical Burst Switching*

1. INTRODUCTION

Optical Burst Switching (OBS) [1] is a type of optical switching paradigm that harnesses the full potentials of wavelength division multiplexing (WDM) technology to transport large volume of data. It draws its good features from both the Optical Circuit Switching (OCS) and Optical Packet Switching (OPS) [2, 3]. In OBS, the packets of data to be transmitted are first assembled into a unit known as a burst at the ingress (edge) node. Following the assembly process, a special packet known as the control packet is generated and transmitted into the network to reserve resources for its burst [1]. During this period, the burst waits at the ingress node until after a period of time known as the offset time before it is transmitted into the network. The control packet is forwarded using a signalling technique that is required to establish a connection for the incoming burst. When the control packet succeeds in establishing a connection, its burst simply cuts-through the intermediate nodes without any optical-electrical-optical (O-E-O) conversion in the core node [1, 4].

Figure 1 depicts an OBS architecture which consists of edge and core nodes that are interconnected by optical links. The edge nodes perform burst assembly/disassembly, offset-time computation, signalling, and routing and wavelength assignment while contention resolution and scheduling functions are performed at the core nodes [5]. Several authors have conducted comprehensive reviews of the different functional parts of OBS such as, burst assembly [4], burst contention resolution [6], quality of service (QoS) provisioning [7], routing and wavelength assignment [8] and core node scheduling [9].

Burst contention is a major issue in OBS [10] and it occurs when two or more bursts request for the same resource at the same time. Contention affects the performance of OBS networks either directly in the form of high burst loss or indirectly in the form of increased delay. Additionally, the network throughout is greatly affected when the contention on the network is high. For these reasons, several contention resolution mechanisms such as wavelength converters, optical buffering in the form of fibre delay lines (FDL) and deflection routing have been proposed in OBS literatures, however, all of them are implemented in the core of the network and they only react when contention occurs. Furthermore, wavelength converters and FDLs are either too expensive, technologically immature [10] or even bulky as in the case of FDL. Therefore, it is better to have a proactive contention handling mechanism at the ingress edge node that can prevent or minimize the chances of contention occurrence at the core of the OBS network. Hence, a suitable candidate for this role is the burst assembly function as it does not require additional hardware and it can take advantage of the electronic memory at the edge.

Burst assembly is crucial to the performance of the OBS network [10] as its performance affects the congestion and contention levels of the network which in turn affects the network performance in terms of delay and loss. Therefore, how can burst assembly be used to minimize contention in an OBS network such that burst loss and end-to-end delay is reduced?

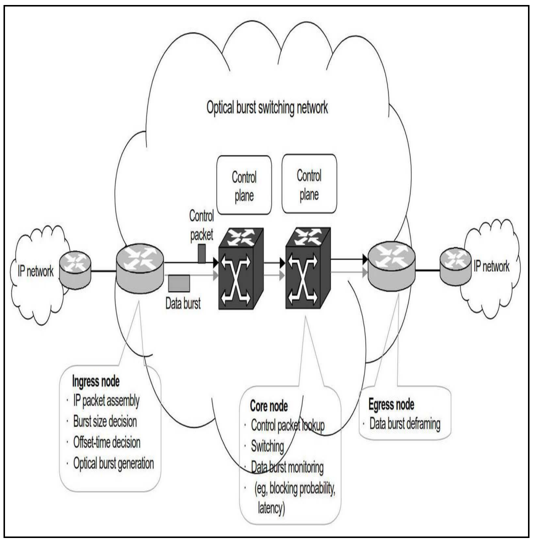

Figure 1: An OBS Network Architecture (Ref. [11]).

In this paper, a novel fuzzy adaptive timer-based burst assembly technique that reduces packet loss and end-to-end (E2E) delay is proposed. Additionally, two sets of fuzzy rules are used to study the effect of different fuzzy rules on the performance of the network.

The organisation of this paper is as follows: Section 2 reviews burst assembly algorithms in OBS. Section 3 describes the proposed fuzzy adaptive timer-based Burst assembly (FATBA) algorithm. Simulation results are discussed in Section 4 and section 5 concludes the paper.

2. RELATED WORKS

The burst assembly process starts at an ingress node upon the arrival of a client data. In the case of an IP packet, the routing module reads the IP header and determines its destination. The packet is passed through the classifier which forwards the packet to the appropriate destination buffer (DB). Upon the arrival of the first bit of data at any empty DB, a timer counter, burst length counter, or both are initialized. The counters will continue counting until either the time threshold or burst length threshold is reached. When either threshold is reached, a burst and its corresponding control packet (BCP) are generated and all threshold counters are re-initialized again. The BCP is sent ahead of its burst to make the necessary reservations while the burst waits at the ingress node output buffer. After a period of time also known as the offset time, the burst is then transmitted into the OBS network. A functional model of an ingress router is shown in figure 2.

There are two basic burst assembly algorithms in OBS: the Timer [12] and Length or Threshold-based [13] algorithms. The timer burst assembly (TBA) algorithm uses a timer as a threshold to assemble burst. Similarly, the threshold-based algorithm uses the number of packets as a threshold for burst generation. When using the length algorithm, packets experience high E2E delay at low loads, and at high loads, bursts of fixed sizes are rapidly generated. Similarly, at low and high loads respectively, the timer algorithm generates small and very large bursts. Therefore, both algorithms increase the burst loss ratio at high loads. Additionally, the thresholds of the basic algorithms are fixed. For the aforementioned reasons, the Min-burst-Length-Max-Assembly-Period (MBMAP) algorithm [14] was proposed to address the limitations of the basic algorithms. MBMAP is a hybrid algorithm that uses both timer and length thresholds for burst generation. Although, MBMAP addressed the issues of the basic algorithms, its thresholds are not adaptive to the incoming traffic. Refs. [15] and [16] have proposed an adaptive hybrid burst assembly algorithm that adjusts the timer and size thresholds using the congestion information of links incident to the ingress node. Even though, the algorithms are adaptive, additional delay is still incurred due to the large burst sizes that are generated.

The threshold-based mixed assembly algorithm [17] aggregates packets of different classes into the same burst. High priority packets are stored at the head while low priority packets are stored at the tail burst. Burst segmentation is used to segment the tail of an already scheduled burst whenever contention occurs. This algorithm still suffers from the drawbacks of the length threshold algorithm. The works of Refs. [18] and [17] are similar except that the high priority packets in Ref. [18] are placed at the tail of the burst. However, the algorithm in Ref. [18] suffers from out-of-order packet delivery problem which increases delay.

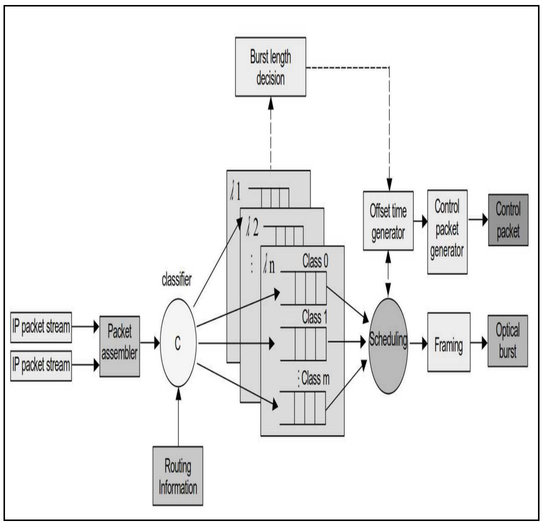

Figure 2: A Functional Model Of An Ingress Node (Ref. [11]).

The adaptive classified cloning and aggregation scheme (ACCS) [19] aims to provide better performance in terms of loss and end-to-end delay for high priority traffic. ACCS uses network loss-rate information to adaptively adjust the hybrid burst assembly thresholds. However, at high loads, ACCS cannot handle the input traffic due to the limitation imposed by the bandwidth at the outgoing link of the edge node. Hence, low priority packets are dropped. Ref. [20] proposed a burst assembly mechanism that aggregates different classes of packets into the same burst. High priority packets are placed at the middle while low priority packets are placed towards the ends of the burst. Both head and tail segmentations are used when contention occurs. Although, this approach reduces the average E2E delay of packets, its implementation is complex, because, the assembler needs to keep additional information about the positions of high and low priority packets within a burst in order for the segmentation mechanism to be effective.

The fuzzy adaptive threshold algorithm (FAT) [21] is based on the length threshold assembly algorithm. FAT uses fuzzy logic together with the incoming traffic information to intelligently adjust the threshold of the assembler. Although, FAT is intelligent in adapting the length threshold, packets and burst may still suffer from high delay depending on how often the fuzzy controller is executed. Refs. [22-24] have proposed several traffic prediction-based burst assembly algorithms that reduce packet E2E delay. These algorithms use prediction techniques to calculate the expected length or timer threshold of the assembler. The estimated length is then included into the control packet which is then transmitted immediately to make an early reservation before the burst generation completes. This approach improves the E2E delay performance; however, poor resource utilization may occur if the generated burst size is higher or lower than the predicted length.

From the above review, it can be concluded that current assembly algorithms still suffer from high E2E delay and loss. Therefore, this paper proposes a new assembly algorithm to address these issues in OBS network.

3. FUZZY ADAPTIVE TIMER-BASED BURST ASSEMBLY ALGORITHM

In TBA algorithm, the timer threshold is a fixed value and it is used to generate a burst when the threshold is reached. Also, TBA does not consider the dynamic nature of the incoming traffic in order to adjust the timer threshold. Hence, the proposed Fuzzy Adaptive Timer Burst Assembly (FATBA) takes into account the offered load and the bandwidth of the output channel in order to adjust the threshold of TBA. Therefore, FATBA is modelled as a fuzzy logic control [25] process in which the control variables are the timer and offered load. The offered load (or load) is the traffic aggregated at the assembler within a time interval. The load and timer control variables are the inputs to the fuzzy logic controller (FLC), and a new timer value is the output. The FLC contains the inbuilt intelligence which is stored as fuzzy rules. These rules are used to adapt the threshold value of the underlying TBA algorithm using the input fuzzy values. The FLC is executed once at the end of every cycle. A cycle is defined as a period of time set by a micro-timer. The micro-timer is independent of the timer threshold of the underlying TBA. Similarly, another independent timer also known as a macro-timer is also executed periodically to reset the load variable. Therefore, whenever the FLC is executed, it produces a new threshold value for the TBA. The new value will be used for the next cycle of the burst assembly process. Figure 3 shows the pseudo-code of the fuzzy adaptive timer burst assembly algorithm. The fuzzy rules are developed based on expert knowledge, and they are stored in the FLC in the form of: **IF** (*Load* **AND** *Timer*) **THEN** (*newTimer*). The Triangular membership function is used for all fuzzy terms as shown in Figure 4. The fuzzy terms set for Timer and newTimer variables are short (S), Average (A), and Long (Ln). Similarly, the terms set for Load variable are low (L), Medium (M) and high (H). The complete set of rules used for the study is given in Table 1.

```
$Timer = t$;
$Load = 0$;
$MinBurstLength = m$;
$newTimer = 0$;
$perSecondTraffic = 0$;
$setFuzzyMicroTimer(k)$;
$setFuzzyMacroTimer(K)$;
$setTimer(Timer)$;
while ($IsTrafficStillArriving?$) do
    $perSecondTraffic += packetSize$;
    if ($getTimer() == Timer$) then
        Call $TimerBurstAssembler()$Return $burst$
        $setTimer(Timer)$;
    end if
    if ($getFuzzyMicroTimer() == k$) then
        $setFuzzyMicroTimer(k)$;
        $Load = perSecondTraffic$;
        Call $FuzzyEngine(Timer, Load)$ Return $newTimer$;
        $Timer = newTimer$;
    end if
    if ($getFuzzyMacroTimer() == K$) then
        $setFuzzyMacroTimer(K)$;
        $Load = perSecondTraffic$;
        $perSecondTraffic = 0$;
    end if
end while
```

Figure 3: FATBA Algorithm

Table 1: Fuzzy Rules

No	Load	Timer	newTimer	
			FATBA-1 (Rules 1)	FATBA-2 (Rules 2)
1	L	S	S	A
2	L	A	S	S
3	L	Ln	S	S
4	M	S	A	A
5	M	A	S	A
6	M	Ln	S	A
7	H	S	A	Ln
8	H	A	Ln	A
9	H	Ln	A	A

4. SIMULATION

A network topology with two cores and four edge nodes is used to evaluate the proposed algorithm in Omnet++ simulator [26] using OBSModules [27] and fuzzylite [28]. Table 2 shows the simulation parameters and settings. All links are bidirectional and load is uniformly distributed. The latest available unscheduled channel with wavelength conversion is used at the core node. The proposed algorithm is evaluated against TBA using the following performance metrics; Average packet queuing delay, burst and packet E2E delays, average burst size, average number of burst sent and packet loss ratio (PLR).

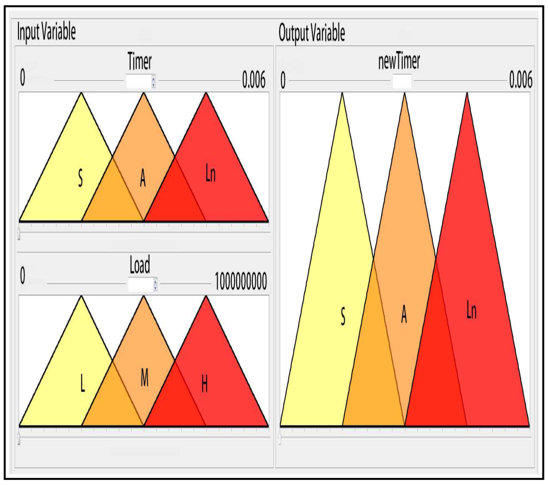

Figure 4 : Linguistic Variables and Membership Functions

Table 2: Simulation Parameters and Values

No.	Parameter	Value
1	Channels per link	4
2	Channel Bandwidth	1 Gbps
3	Packet Inter-arrival	Exponential
4	Packet Size	1250 Bytes
5	Fuzzy Micro Timer	0.1 s
6	Fuzzy Macro Timer	1 s
7	BCP Proc. Time	10 us
8	Min. Timer	0.001 s
9	Max. Timer	0.006 s
10	Init. Timer Threshold	0.003 s
11	Min. Burst Length	1500 bytes
12	T-Norm	algebraic-product
13	S-Norm	algebraic-sum
14	Activation	minimum
15	Accumulation	maximum
16	Inference Engine	Mamdani
17	Defuzzification	Centre of Gravity

The following figures shows simulation results of TBA and the two rules of FATBA (FATBA-1 and FATBA-2).

Figure 5 shows the packet average queuing delay against offered load for TBA, FATBA-1 and FATBA-2. The proposed algorithm exhibits lower queuing delays when compared to the TBA. Furthermore, FATBA-1 has the lowest queuing delay because its fuzzy rules set (see Table 1) produces a timer threshold that satisfies the burst generation condition based on the available load.

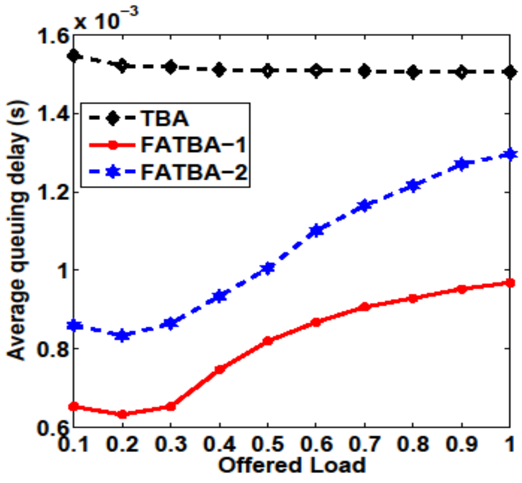

Figure 5: Packet Average Queuing Delay

Figure 6 depicts the average burst E2E delay against offered load for TBA, FATBA-1 and FATBA-2. As the load increases, TBA generates larger bursts that have longer transmission duration. Moreover, FATBA-1 and FATBA-2 are able to minimize the delay by generating bursts of average sizes that have shorter transmission duration. However, FATBA-1 outperforms FATBA-2 because its fuzzy rules set satisfy the generation of smaller bursts that have shorter transmission delay.

Figure 7 shows the plot of the average packet E2E delay against offered load. The packet E2E delay performance is a function of the average packet queuing delay, burst E2E delay and burst disassembly delay. The packet E2E delays of both FATBA-1 and FATBA-2 are still lower than that of TBA because, both FATBAs have lower average queuing and burst E2E delays.

The results of average PLR against offered load for TBA, FATBA-1 and FATBA-2 are shown in Figure 8. All the algorithms exhibit a similar pattern of loss for loads between 0.1 and 0.8. However, from loads 0.9 to 1, FATBA-1 generates smaller bursts that have lower blocking probabilities than those generated by both FATBA-2 and TBA. Hence, FATBA-1 has the lowest PLR as shown in Figure 8.

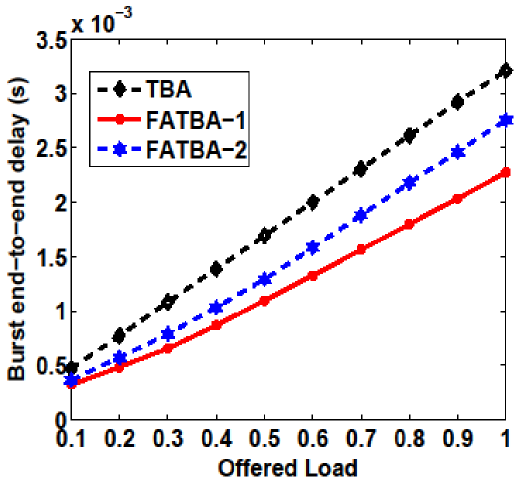

Figure 6: Burst End-To-End Delay

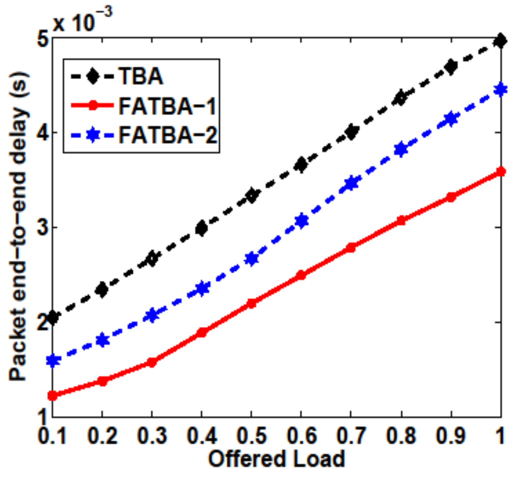

Figure 7: Packet End-To-End Delay

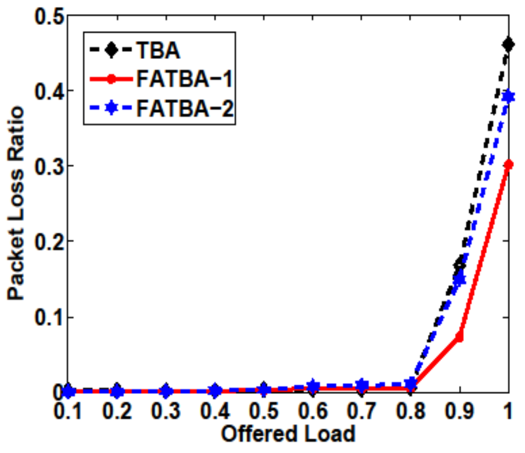

Figure 8: Packet Loss Ratio

Figure 9 compares the average size of bursts generated against offered load. The figure depicts that, for all loads, FATBA-1 generates the shortest bursts followed by FATBA-2 and TBA. This shows how the different fuzzy rules set adapts the timer threshold to produce different timer values in order to generate bursts of variable sizes that also depend on the available load.

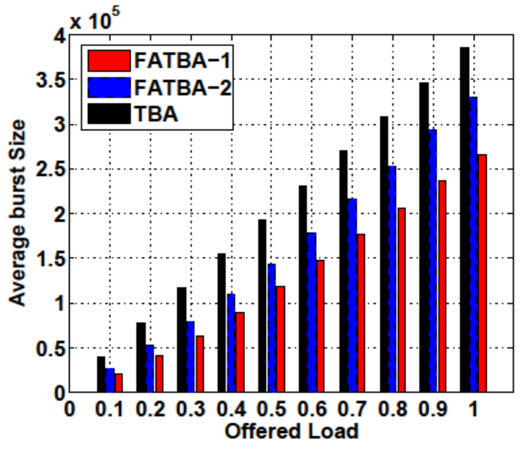

Figure 9: Average Burst Size

The plot of the average number of burst generated against offered load is shown in Figure 10. The figure shows that, the number of bursts generated by TBA is fairly constant for all loads. However, the number of bursts generated by FATBA-1 and FATBA-2 is higher than those sent by TBA. FATBA-1 generates the highest number of burst for every load because it assembles shortest bursts. Similarly, the same argument holds true for FATBA-2 except that, FATBA-2 fuzzy rules set allows bigger bursts to be generated than those in FATBA-1.

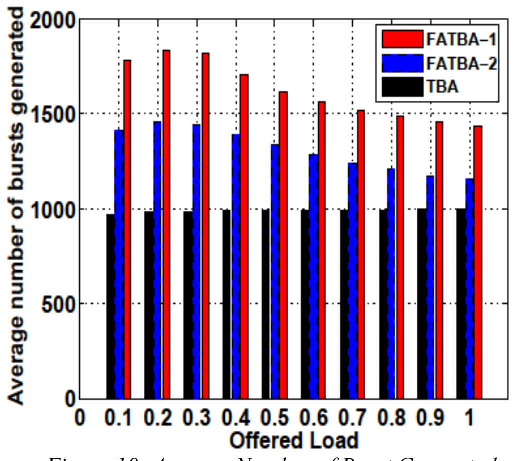

Figure 10: Average Number of Burst Generated

The above results show that the incorporation of fuzzy logic control into the timer-based assembly algorithm has added flexibility and intelligence into the assembly process. Hence, FATBA can be considered as a suitable candidate for real-time applications that are delay sensitive. However, FATBA's flexibility has added an additional control overhead to the OBS network.

5. CONCLUSION

In this study, a fuzzy logic based burst assembly algorithm that minimizes loss and delay is proposed. The proposed algorithm uses fuzzy logic and the input traffic to dynamically adjust the threshold of the timer-based burst assembly algorithm. Furthermore, two sets of fuzzy rules are used to evaluate the proposed algorithm against the traditional timer-based assembly algorithm. The evaluation was carried out through computer simulation and results have shown that the new algorithm outperforms the traditional algorithm in terms of loss and delay. Additionally, this study shows how different fuzzy rules affect the performance of the network. In the future, quality of service is to be incorporated into the proposed algorithm to support different types of traffic that will further improve network performance.

ACKNOWLEDGEMENT

This Research is supported by Universiti Technologi Malaysia and Ministry of Education, Malaysia through Commonwealth Scholarship and Fellowship Plan 2011 and Fundamental Research Grant Scheme (FRGS:R.J130000.7828.4F324).

REFRENCES:

[1] Q. Chunming and Y. Myungsik, "Optical Burst Switching (OBS) - A New Paradigm for an Optical Internet," *Journal of High Speed Networks*, vol. 8, pp. 69-84, 1999.
[2] V. M. Vokkarane and J. P. Jue, "Segmentation-based nonpreemptive channel scheduling algorithms for optical burst-switched networks," *Journal of Lightwave Technology*, vol. 23, pp. 3125-3137, 2005.
[3] C. Yang, Q. Chunming, and Y. Xiang, "Optical burst switching: A new area in optical networking research," *IEEE Network*, vol. 18, pp. 16-23, 2004.
[4] H. Kaur and R. S. Kaler, "Burst Assembly and Signaling Protocols in OBS," presented at the Proceedings of National Conference on Challenges and Opportunities in Information

Technology COIT-2007 RIMT-IET, Mandi Gobindgarh, India, 2007.

[5] M. Maier, *Optical Switching Networks*, 1 ed. New York: Cambridge University Press, 2008.

[6] X. Yu, J. Li, X. Cao, Y. Chen, and C. Qiao, "Traffic statistics and performance evaluation in optical burst switched networks," *J. Lightw. Technol,* vol. 22, pp. 2722-2738, 2004.

[7] T. Tachibana and S. Kasahara, "QoS-guaranteed burst transmission for VoIP service over optical burst switching networks," *Journal of Optical Networking,* vol. 6, pp. 991-1002, 2007.

[8] C. Yahaya, M. S. Abd Latiff, and A. B. Mohamed, "A review of routing strategies for optical burst switched networks," *International Journal of Communication Systems,* vol. 26, pp. 315-336, 2013.

[9] J. Li, C. Qiao, and Y. Chen, "Recent progress in the scheduling algorithms in optical-burst-switched networks [Invited]," *Journal of Optical Networking,* vol. 3, pp. 229-241, 2004.

[10] X. Yi-Yuan and Z. Jian-Guo, "An intelligent segmented burst assembly mechanism in optical burst switching networks," *Chinese Physics Letters,* vol. 25, p. 2535, 2008.

[11] S.-y. Oh, H. H. Hong, and M. Kang, "A Data Burst Assembly Algorithm in Optical Burst Switching Networks," *ETRI Journal,* vol. 24, pp. 311-322, 2002.

[12] A. Ge, F. Callegati, and L. S. Tamil, "On Optical Burst Switching and Self-Similar Traffic," *IEEE Communications Letters* vol. 4, pp. 98-100, 2000.

[13] V. M. Vokkarane, K. Haridoss, and J. P. Jue, "Threshold-based burst assembly policies for QoS support in optical burst-switched networks," presented at the The Convergence of Information Technologies and Communications, 2002.

[14] X. Cao, J. Li, Y. Chen, and C. Qiao, "Assembling TCP/IP packets in Optical Burst Switched Networks," presented at the IEEE GLOBECOM'02, Taipei, China, 2002.

[15] B. Kantarci and S. Oktug, "Adaptive Threshold Based Burst Assembly in OBS Networks," presented at the IEEE Canadian Conference on Electrical and Computer Engineering, 2006.

[16] A. Gupta, R. S. Kaler, and H. Singh, "Investigation of OBS assembly technique based on various scheduling techniques for maximizing throughput," *Optik - International Journal for Light and Electron Optics,* vol. 124, pp. 840-844, 5// 2012.

[17] Z. Zhang, J. Luo, Q. Zeng, and Y. Zhou, "Novel threshold-based burst assembly scheme for QoS support in optical burst switched WDM networks," in *Performance and Control of Next-Generation Communications Networks*, 2003, pp. 250-256.

[18] Z. Zhang, F. Cheng, J. Wang, J. Luo, Q. Zeng, and X. Xuan, "A new burst assembly and dropping scheme for service differentiation in optical burst switched networks," in *Optical Transmission, Switching, and Subsystems*, 2004, pp. 236-245.

[19] S. Askar, G. Zervas, D. K. Hunter, and D. Simeonidou, "Adaptive classified cloning and aggregation technique for delay and loss sensitive applications in OBS networks," in *Optical Fiber Communication Conference and Exposition (OFC/NFOEC), 2011 and the National Fiber Optic Engineers Conference*, 2011, pp. 1-3.

[20] A.-h. Guan, F. Hu, and W.-c. Li, "A new composite assembly mechanism for supporting QoS in OBS networks," *Optoelectronics Letters,* vol. 10, pp. 55-58, 2014/01/01 2014.

[21] J.-r. YANG, S.-l. JIA, and G. WANG, "Burst assembly algorithm based on fuzzy-adaptive-threshold " *Journal of Harbin Engineering University,* vol. 6, p. 013, 2007.

[22] H.-l. Liu and S. Jiang, "A mixed-length and time threshold burst assembly algorithm based on traffic prediction in OBS network," *Int. J. Sensing, Computing & Control,* vol. 2, pp. 87-93, 2012.

[23] X. JIANG, N. ZHU, and L. YUAN, "A Novel Burst Assembly Algorithm for OBS Networks Based on Burst Size and Assembly Time Prediction," *Journal of Computational Information Systems,* vol. 9, pp. 463-475, 2013.

[24] K. Seklou, A. Sideri, P. Kokkinos, and E. Varvarigos, "New assembly techniques and fast reservation protocols for optical burst switched networks based on traffic prediction," *Optical switching and networking,* vol. 10, pp. 132-148, 04/01 2013.

[25] C. C. Lee, "Fuzzy logic in control systems: fuzzy logic controller. II," *IEEE Transactions on Systems, Man and Cybernetics,* vol. 20, pp. 419-435, 1990.

[26] A. Varga and R. Hornig, "An overview of the OMNeT++ simulation environment," in

Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops, 2008, p. 60.

[27] F. Espina, J. Armendariz, N. Garc, D. Morat, M. Izal, and E. Maga, "OBS network model for OMNeT++: a performance evaluation," presented at the Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques, Torremolinos, Malaga, Spain, 2010.

[28] J. Rada-Vilela. (2013, 01-03-2013, URL: http://www.fuzzylite.com). *fuzzylite: A fuzzy logic control library written in C++*.