Gnutella-Based P2P Applications for SDN over TWDM-PON Architecture

Anish Sah, I-Shyan Hwang, Ardian Rianto, Andrew Fernando Pakpahan, and Andrew Tanny Liem

***Abstract*—Demands for Peer-to-peer (P2P) applications are rapidly growing to become the most popular bandwidth consumers in the world. Among the various P2P applications, Gnutella is the most popular unstructured P2P networks allowing the sharing of files at a very high rate. TWDM-PON has been regarded as the promising solution to meet the higher bandwidth demands next-generation passive optical network (NG-PON2). It provides flexibility to support multiple services to multiple organization on the same fiber. SDN (software-defined networking) is the emerging technology that decouples the control and data plane and centralized the network intelligence at one place. As a result, the operators get programmability, automation and network control to manage a network that freely adapts the changes needed to the business. In this paper, a new Gnutella application for SDN over TWDM-PON architecture is proposed that the OLT and ONU are capable of handling the Gnutella traffic generated by Gnutella applications, and the Gnutella Engine Manager is controlled by SD-controller. The proposed mechanism is able to reduce the huge bandwidth waste caused by flooding controlling messages, guarantee the success of query and also localize the Gnutella inter and intra traffic between PON which improve the quality of services (QoS) in terms of the mean packet delay, jitter, system throughput and packet dropping.**

***Index Terms*—P2P, Gnutella, TWDM-PON, SDN, QoS.**

I. Introduction

P2P networking has generated tremendous interest worldwide among both internet and computer network professionals. P2P file sharing systems have become the single most popular class of internet application in this decade. Numerous businesses and websites have promoted P2P technology as the future of Internet networking. Although they have actually existed for many years, P2P technologies promise to radically change the future of networking [1]. P2P network is a distributed network in which each node has the equal capability and ability to exchange the information with each other directly. The P2P system can be categorized into two parts - *unstructured* and *structured* [2]. In P2P, the 'peers' are the computers which are connected to each other via the internet where files can be shared directly between the systems on the network without the need of a central server; so, we can say that each node acts as a server. Based on the unstructured architecture, the P2P network can be categorized into *Pure Decentralized*, *Hybrid Decentralized*, and *Partially Centralized* [3]-[5]. In a Pure Decentralized network, all nodes in the network act as servers and/or client, while Hybrid Decentralized server facilitates the interaction between peers by maintaining directories of the shared files stored on the respective PCs of registered users to the networks. In a partially centralized system, the basis is the same as with purely decentralized systems. However, some of the nodes assume a more important role than the rest of the nodes and acting as local central indexes for files shared by local peers [6], [7]. The main purpose of the P2P design is for peers to correspond on the internet, without the need for new protocol on switches and router in the internet core. In P2P computing, nodes organize themselves as an overlay network, in which packet transmission on each of the overlay links uses standard Internet protocols, that is Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) [8].

Gnutella is a large P2P network which was the first decentralized P2P network of its kind to allow users to share resources, leading to other, later networks adopting the model [9], [10]. Gnutella is the top P2P file sharing system in the world and it is the most popular peer-to-peer application that it is an open source application. Gnutella protocol comes in various versions: Gnutella 0.4 and Gnutella 0.6. The number of active node in Gnutella network 0.4 is 5, whereas the number of maximum hope is 7. In Gnutella protocol version 0.4, the concept of servents are used and performs tasks normally associated with both clients and servers. Gnutella protocol version 0.6 uses a two-level hierarchy: Ultrapeers (UPs) and Leaf Nodes (LNs). UPs can connect to the high capacity links and have a large amount of processing power. LNs maintain a single connection to their UPs, and the UP maintains 10-100 connections, one for each LN and 1-10 connections to other nodes [11]. In Gnutella, there are five main message-controlling mechanisms Ping, Pong, Query, Query Hit and Push. According to [12], they monitor Gnutella traffic in which Ping, Pong, and Query message altogether account for more than 95% of traffic, where the QUERY is 54.80%, PONG is 26.90%, PING is 14.80% and QUERY-HIT is 2.80%, respectively.

The original search algorithm used in a Gnutella system [13] was the flooding search algorithm that brings the major issue of low success rate and a huge waste of bandwidth. In [14], a new search algorithm called 'AntSearch' was proposed to reduce the network traffic caused by the flooding algorithm. According to [15], they improve the current routing algorithm based on the Ant algorithm for the optimal routing. In the proposed routing search algorithm, they

Manuscript received November 25, 2019; revised February 12, 2020.

Anish Sah, I-Shyan Hwang, and Ardian Rianto are with the Department of Computer Science and Engineering, Yuan Ze University, Chung-Li, 32003, Taiwan (e-mail: ishwang@saturn.yzu.edu.tw).

Andrew Fernando Pakpahan is with the Department of Computer Science, Universitas Klabat, Manado, 95371, Indonesia.

Andrew Tanny Liem is with the Department of Information Technology, Universitas Advent Indonesia, Bandung 40559, Indonesia.

DOI: 10.7763/IJCTE.2020.V12.1260

avoided the randomization and blindness of message relay service by adding a constrained condition. Few of the research [16] compared the strength and weaknesses of three algorithms of Gnutella P2P protocol namely Flood, Random Walk, and Random Walk with Neighbors and they proposed a new search method based on the experiment. A new hierarchical architecture in [17] is proposed for Gnutella network by categorizing the nodes as client-nodes and super-nodes which was able to make the network scale, by reducing the number of nodes on the network involved in message handling and routing as well as reducing the actual traffic among them.

Most of the recent researches for Gnutella focus on decrease the flooding problem and reducing the network traffic to minimize the waste of bandwidth. Therefore, we can easily understand the current researches target is to control the network utilization of P2P application while minimizing the inter-Internet Service Provider (ISP) traffic and improving the quality of service (QoS). In optical access networking, locality awareness in the neighbor has become one of the most promising solutions to decrease the amount of inter-ISP P2P traffic [18] which integrates peering mechanism into the network infrastructure which greatly simplifies the implementation of local policies. In the access network, the PON system has to be an attractive technology that can offer high bandwidth with low latency. Each PON system architecture consists of a central Optical Line Terminal (OLT) which is connected with multiple Optical Network Units (ONUs) and one optical splitter [19], [20]. To avoid the data collision between the ONUs, the IEEE 802.3 ah has developed a standard, called Multi-Point Control Protocol (MPCP) [21]-[23]. Dynamic bandwidth allocation (DBA) allocates the appropriate bandwidth to each ONU, which is a method for assigning bandwidth dynamically based on the queue state information received from ONUs. In our previous work [24], we tried to reduce the inter- and intra- traffic in the PON and ISPs, and also improve the QoS by localizing the intra-ISP traffic. Likewise, in [25], we use the application-aware mechanism to store some popular context in the local system by supporting local content; moreover, we tried to decrease the amount of inter-ISPs traffic by localizing the intra-ISP traffic in which we designed new ONU mechanism (patching and caching) to reduce the resource consumption and provide more downstream bandwidth without be buffered and scheduled in the downstream direction by the OLT.

There are various PON standards, started form EPON, G-PON, XG-PON1, and NG-PON2. In G-PON standards, the bandwidths supported were downstream 2.5G and upstream 1.25G; whereas in XG-PON1 standard the bandwidth capacities were downstream 10G and upstream 1.25G or 5G. Finally, the NG-PON2 bandwidth can support up to 40G [26], [27]. Among all of the aforementioned proposals, Time and Wavelength Division Multiplexed Passive Optical Network (TWDM-PON) technology are chosen by the telecommunication industry for implementation of NG-PON2 by the FSAN community in April 2012. TWDM-PON increases the aggregate PON rate by stacking XG-PONs via multiple pairs of wavelengths.

Software Defined Network (SDN) has gained a lot of attention in the last few years starting from 2012 [28] The Open Network Foundation (ONF) introduced a new concept of Networking whereby the control and data planes are decoupled, networking intelligence and state are logically centralized, and the underlying networking infrastructure is abstracted from the applications. The distributed control and transport network protocols running inside the routers and switches are the key technology that allows information in the form of digital packets to travel around the world [29]. The SDN is considered as the optimal choice for the Next-generation PON with the advantage of flexible and centralized control capacity It aims to lead the centralized programmable model of the network in which the OpenFlow protocol used to adapt the SDN mechanism into a network. OpenFlow is based on an Ethernet switch with internal flow-table and the standardized interface to add and remove flow entries [30]. Implementation of SDN over PONs [31]-[33] can reduce the energy consumption of TWDM-PONs, provide dynamic flex-grid wavelength circuit creation, and manage the traffic.

In this paper, by taking advantage of SDN and OpenFlow protocol in TWDM-PON, a new Gnutella-Applications for SDN over TWDM-PON architecture is proposed, where we used the concept of the partially centralized architecture of P2P shown in Fig. 1. By implementing this architecture, we are able to reduce the huge bandwidth waste caused by flooding controlling message and guarantee the success of query by reducing the dropping, localizing the intra-traffic and accommodate a large number of users, and to guarantee the network scalability by the improvement of QoS.

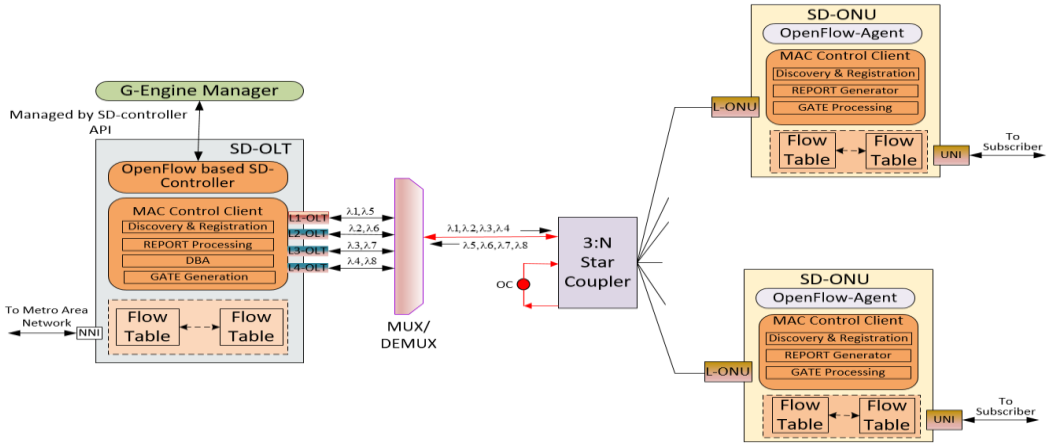


Fig. 1. Proposed Gnutella-Application based file sharing architecture for SDN.

The rest of the paper is organized as follows. Section II introduces the proposed system architecture and mechanism including OLT, ONUs and G-DBA Operations. Section III conducts the overall system performance evaluation in terms of packet delay, jitter, throughput, and packet dropping. Section IV gives the conclusion of the paper.

II. Proposed System Architecture and Mechanism

This section describes the proposed Gnutella-Application for SDN over TWDM-PON architecture. In the proposed architecture, the open-flow based SD-controller is capable of communicating with the Gnutella-Engine Manager (G-Engine Manager) through North Bound APIs. The OpenFlow based SD-controller manages the traffic flows between the ONUs and OLT, and it takes benefits of flow tables in ONUs containing the flow entry of each packets traveling in the network to reduce the waste of bandwidth done by Gnutella application.

A. System Architecture

we proposed a hierarchical Gnutella network shown in Fig. 1 by categorizing the nodes as Lead Nodes and Ultra Nodes [16] that the G-Engine Manager as an Ultra Node and ONUs as a Leaf Node. The architecture consists of five main components, which are Gnutella-OLT (G-OLT), Gnutella-ONU (G-ONU), G-Engine Manager, 3:N Star Coupler (3:N SC) and MUX/DEMUX. The 3:N SC broadcasts the downstream traffic tuned at the wavelength 1-4, upstream traffic tuned at the wavelength 5-8, and the intra traffic tuned to wavelength λ_{p2p}.

B. Proposed OLT Architecture

Fig. 2 shows the details of Software-Defined OLT (SD-OLT) architecture which contains the flow table, buffered manager and SD-controller. The packet send by ONUs is being received by the receiver (Rx) which is then classified by packet classifier according to the packet types. The packet classifier contains CoS (Class of Service) which classifies the traffic according to the packet types, Expedited Forwarding (EF) traffic, Assured Forwarding (AF) traffic and Best Effort (BE) traffic; and ToS (Type of Service) classifies the types of services. In our case, ToS will classify the GT (Gnutella Traffic) from the BE traffic. If SD-controller found some packets that belong to Gnutella Traffic, first it will check whether the requested packet is already in the same PON. This work is done by checking the flow entry of OLT and ONUs, and if it found that the requested packet is already in ONUs, the SD-Controller will update the SD-Agent. After that, the ONUs can request for the time slots to process the request packets. If the requested packet is not found in the flow table entry of ONUs, it will redirect the packet request to the G-Engine Manager for further processing. The G-Engine Manager consists of the packet processing engine, processing unit, storage, and buffer. The packet processing engine is equipped with the Query Routing Table (QRT) and the Distributed Hash Table (DHT) [34]. QRT is used to maintain the routing table of each user where the routing table contains a number of packets shared by each user. If the packet is not found in PON or Local LAN, it will redirect the message out of PON. DHT is used to maintain the index value of shared file by each ONUs, where the key value is generated by each node and makes their own DHT, and G-Engine Manager maintains and updates the DHT tables.

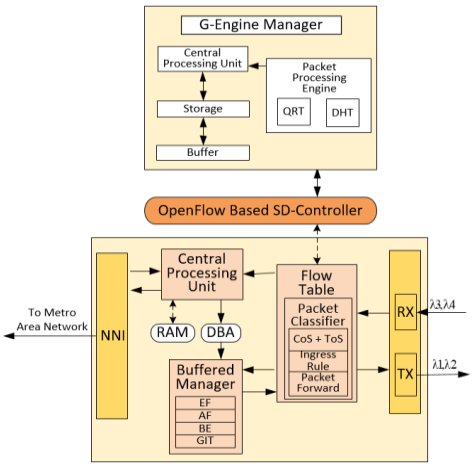


Fig. 2. Detailed software-defined OLT.

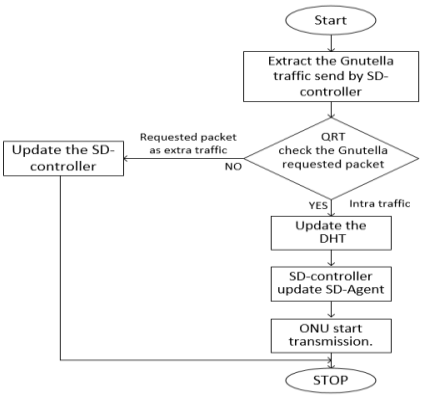


Fig. 3. Detailed operations of Gnutella.

Fig. 3 shows the detailed operations of G-Engine Manager, after classifying the packet by using packet classifier, the SD-controller will send all the packets that need to be processed by Gnutella Engine Manager. The Gnutella Engine Manager contains the QRT and DHT, so it can store all the historical Gnutella activity that is done in Local LAN and PON. It keeps records of all the packets that are being shared by ONUs users according to packet types including its index values which are then stored in DHT.

Once the packet is sent by SD-controller to G-Engine Manager, the QRT will check if the requested packet is being previously used by ONUs user. If QRT found the requested packet is not being previously used in the local users or PON, it will request the packet from external supernodes. If the QRT found that the requested packets were previously used by ONUs users, it will update the DHT with new index values together with all the necessary user information. After that, it updates the SD-controller and the OLT flow table. The SD-controller will update the SD-Agent and ONUs flow table with the information provided by G-Engine Manager. Once the SD-Agent updates the ONUs, users can request the timeslot for the transmission of the data.

Both OLT and ONUs can directly process the Report and Gate message. In our OLT architecture, we have modified the

Ingress rule; if OLT receives any packet that belongs to Gnutella traffic it will forward to G-Engine manager by SD-controller for further processing.

C. Proposed ONU Architecture

Fig. 4 shows the detailed ONU architectures. The user interacts with UNI and subscribes to the network, whereas the packet classifier has three modules, which enables the ONUs to classify the user traffic based on different parameters. According to the Ingress, if the packet classifier found that the packet belongs to Gnutella traffic, it will check the flow table entry, and then check if the destination is on the same ONU or different ONUs. If in the same ONU, it redirects the packet, but if the end user request is in other ONU, it is sent to the queue manager for future processing. The intra traffic request is handled by star coupler by redirecting the traffic between the ONUs. If the request is not in the same PON, it is sent to G-Engine Manager for further processing

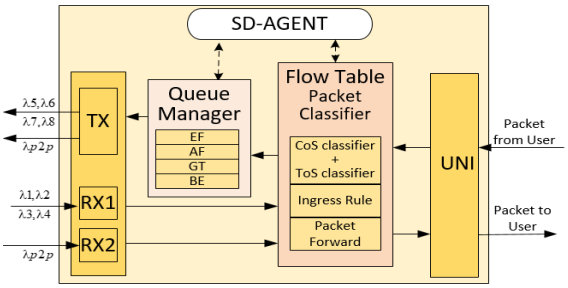


Fig. 4. Detailed-software defined ONU.

The flow table is responsible for all records of flow entries which contain the header files, counters, and actions. The queue manager has four queues, Expedited-Forwarded (EF), Assured-Forwarding (AF), and Best-Effort (BE), and one other queue is referred to as GIT (Gnutella intra-Traffic). The SD-Agent enables the SD-controller mechanism for the SD-ONUs to connect between the OLT and ONUs. The objective is to reduce the huge waste of bandwidth caused by query message.

D. Gnutella-Dynamic Bandwidth Allocation (G-DBA)

Fig. 5 shows the details of the proposed new DBA scheme to handle the Gnutella-based traffic. The proposed scheme supports intra PON traffic with four priority queues at each ONUs, which are EF, AF, GIT and BE respectively. Once the OLT receive the Report message, the OLT will calculate the time slots according to the traffic types. The standard REPORT frame format has 8 queues. In our proposed architecture, we only used 4 of them. Queue#0 represents EF traffic, Queue#1 represents the AF traffic, Queue#2 represents the GT (Gnutella Traffic) and the Queue#4 represent the BE traffic. Here, the G-DBA will assign the timeslots according to the Priority Queue and available bandwidth. First it will allocate the bandwidth to EF traffic, then checks remaining timeslot and allocates to AF traffic. After EF and AF traffic it will allocate the GT Traffic and finally the remaining timeslot to the BE traffic. Once the time slots are being calculated for all traffic, there will be a message containing start time, length, and wavelength for each traffic for all ONUs.

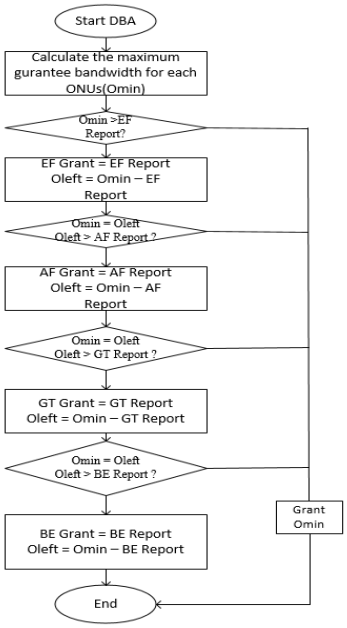


Fig. 5. Proposed G-DBA flow chart.

E. Proposed Gnutella Signaling Operation

The Gnutella signaling operation is shown in Fig. 6. Initially, if a node wishes to participate in the Gnutella network, they would join by finding an initial host to start its first connection. If the ONU is leaf node, the initial host will be G-Engine Manager/Ultrapeers; then, OLT will send the initial Discovery GATE message to all the connected ONUs.

Autodiscovery mode is used to discover and initialize the newly activated ONUs in the network. It is also used to learn round trip delays and MAC address of that ONUs and also assign Logical Link Identification (LLID) parameters for ONUs. The autodiscovery process is implemented in both the OLT and ONUs with four MPCP control messages carried in MAC control frames: GATE, REGISTER_REQ, REGISTER, and REGISTER_ACK. The OLT sends a discovery GATE to all ONUs to create the transmission opportunity for the undiscovered ONU. Undiscovered ONU generates a REGISTER_REQ message that remains buffered until the transmission windows open. Then the REGISTER_REQ is transmitted upstream to the broadcast channel. At last, the OLT replies by sending REGISTER_ACK to finish the autodiscovery process. After the autodiscovery process, the OpenFlow connection between the SD-controller and SD-Agent is established by sending the OFPT_HELLO message on each side. OFPT_ERROR message will be sent if the connection fails. After establishing successful connection, the controller sends an OPFT_FEATURE_REQUEST message which is responded with OFPT_FEATURE_REPLY by the SD-Agent of ONU. After that, the SD-controller setups the OPT_CONFIGURATION in SD-Agent. The handshaking operations between the ONU and OLT begins with sending of the string Gnutella Connect/0.6<CR><LF>, where <CR> is the ASCII code for carriage return and <LF> is the ASCII code for line feed. The OLT responds with the string Gnutella/0.6 <status code> <statues string><CR><LF>. The status <code> follows the HTTP specification with code 200 meaning success [35]. If a client wishes to connect, the client

responds and sends GNUTELLA/0.6 with setting to 200. If not, it will send an error message and closes the TCP connection. The PING message is sent by OLT to all the ONUs to show their presence on the network, and all ONUs responds by PONG message. The PONG message contains the Port Number, IP Address, Number of File Shares and Number of KB Shares, which will be updated in the ONUs Flow Tables and G-Engine Manager by the SD-controller. The query is used to search the distributed network and response to query-hit from ONUs. OFPT_FLOW_MOD is used to update the flow table of SD-Agent.

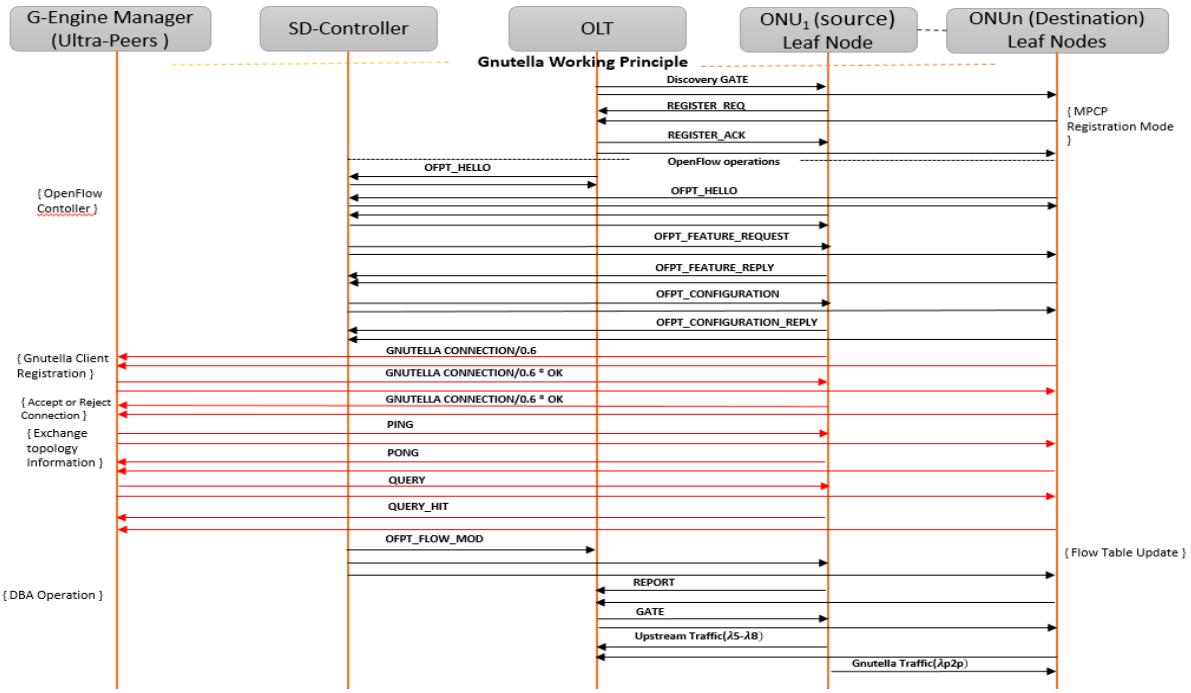


Fig. 6. Gnutella signaling operations.

The GATE message specifies the transmission of start and ends time during which the ONUs can transmit the queued customer traffic upstream to the OLT. If the packet is intra traffic, the SD-Agent communicates with the SD-controller for information about the QRT and DHT then processes the packet and subsequently updates the flow tables.

III. Performance Evaluation

In this section, we demonstrated our result by comparing it with IPACT [36]. Our simulation experiment compares the end-to-end packet delay, jitter, system throughput and packet dropping probability. The system model is set up in the OPNET with OLT and 64 ONUs. The data rate of downstream and upstream both amounts to 4 Gbps and the ONU buffer size is 10 MB. The distance between the ONUs and the OLT is uniform from 10 to 20 km. The traffic models AF and BE are the networks traffics chosen for their self-similarity and long-range dependence (LRD) whereas, the highest priority (i.e., EF traffic) uses poison distribution. Self-similarity and long-range dependence are utilized to generate the highly burst BE and AF traffic with a Hurst Parameter of 0.7, and packet sizes of AF and BE are uniformly distributed between 512 and 12144 bytes while EF packet sizes are constantly distributed at 560 bytes. The Gnutella packet is uniformly distributed between 9600 and 12144 bytes. The proposed scheme support four priority queues at each ONUs, which are EF, AF, GT (Gnutella Traffic) and BE, respectively. The simulation scenario and parameter are shown in Table I and Table II respectively.

TABLE I: Simulation Scenarios

Scenarios	EF	AF	BE	Gnutella Traffic(GT)
Original IPACT-10:30:60	10%	30%	60%	None
Original IPACT- 10:40:50	10%	40%	50%	None
Original IPACT- 10:50:40	10%	50%	40%	None
Case1 - 10:30:60(15%)	10%	30%	51%	9%
Case2 - 10:40:50(15%)	10%	40%	42.5%	7.5%
Case3 - 10:50:40(15%)	10%	50%	34%	6%
Case4 - 10:30:60(25%)	10%	30%	45%	15%
Case5 - 10:40:50(25%)	10%	40%	37.5%	12.5%
Case6 - 10:50:40(25%)	10%	50%	30%	10%

TABLE II: Simulation Parameter

Parameters	Values
Number of OLT	1
Number of ONU	64
OLT-ONU distance(uniform)	10-20 Km
Up/Down link capacity	4 Gbps
Max cycle time	1.0ms, 1.5ms
Guard Time	1 μs
DBA computation	10 μs
Control message length	0.512 μs
ONU buffer size	10 Mb
AF and BE packet Size(bytes)	Uniform(512,12144)
EF packet size(bytes)	Constant(560)

A. Mean Packet Delay

The mean packet delay occurs when the packets reach the ONUs at random time periods. Each packets will have to wait for their allocated time period to transmit the upstream data, where the waiting time is referred to as packet delay which consists of the polling, granting and queuing delays [37]. Fig. 7 shows the improvement of delays in the different scenarios as compared with IPACT. In Case 4 with 1.5ms cycle time, we obtained maximum improvement in delay. At 1.5ms cycle time, the result shows the improvement of 22% for EF delay, 22.5% for AF delay and 49.4% for BE delay whereas for 1.0ms cycle time we demonstrated the improvement of

14.8% for EF delay, 14.9% for AF delay and 30.7% for BE delay, respectively. We can see the improvement because the request does not have to pass through OLT, it can immediately redirect the packet using one extra wavelength (λ_{p2p}). In Cases 3 and 6, there is less delay improvement because the AF traffic value is higher as compared to other cases.

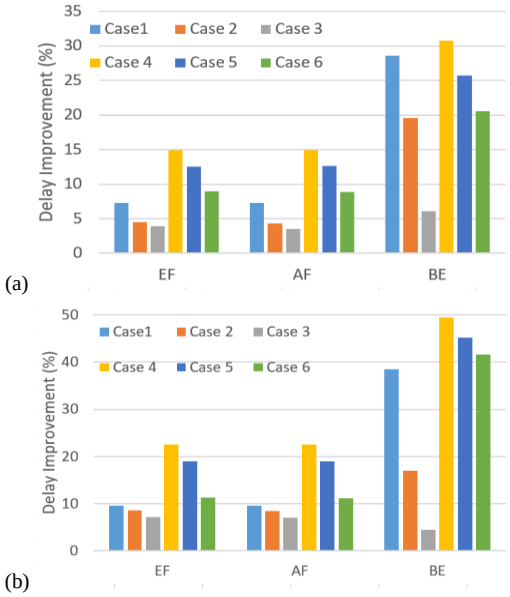


Fig. 7. Average delay improvement with different traffic ratios for (a) 1.0ms cycle time (b) 1.5ms cycle time.

Fig. 8 shows the Gnutella delay between two different cycle times. In 1.0ms cycle time, the high traffic load delay suddenly increased because Cases 3 and 6 have the highest AF traffic than other cases. This means that it needs more timeslot to transfer the data so that subsequently our Gnutella traffic will get the chance to transfer the data as well. Our traffic prioritizes EF, AF, then Gnutella and lastly, BE traffic. In the 1.5ms cycle time, the identical situation was alleviated because the timeslot given by OLT is enough to transfer all including the Gnutella traffic.

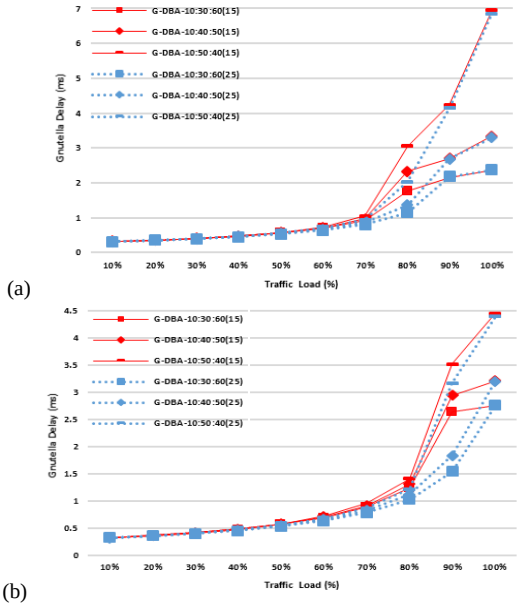


Fig. 8. Gnutella delay with different traffic ratios for (a) 1.0ms cycle time (b) 1.5ms cycle time.

B. Jitter Performance

Jitter is the packet transfer delay variation and it has a significant impact on voice quality. A smaller jitter value is required to deliver better and high-quality voice signal.

Fig. 9 shows the EF jitter different traffic ratios for different scenarios. The proposed EF jitter in the high load (100%) traffic is almost the same as the original IPACT. However, in some scenarios from 50% to 90%, EF jitter is improved in high traffic load as compared to IPACT.

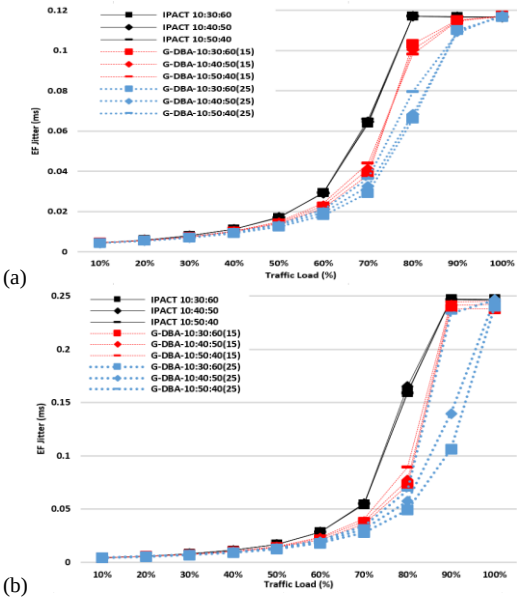


Fig. 9. EF Jitter with different traffic ratios for (a) 1.0ms cycle time (b) 1.5ms cycle time.

Fig. 10 shows the Gnutella jitter performance whereby in Scenarios 3 and 6 at 1.0ms cycle time with traffic load (90% to 100%) suddenly increased because the AF traffic is higher as compared to other scenarios. This causes the remaining traffic to be sent in the next given cycle time. In the case of 1.5ms cycle time, the timeslot given by OLT is enough for ONUs to send the AF traffic, and the same goes for the remaining traffic like Gnutella and BE traffic.

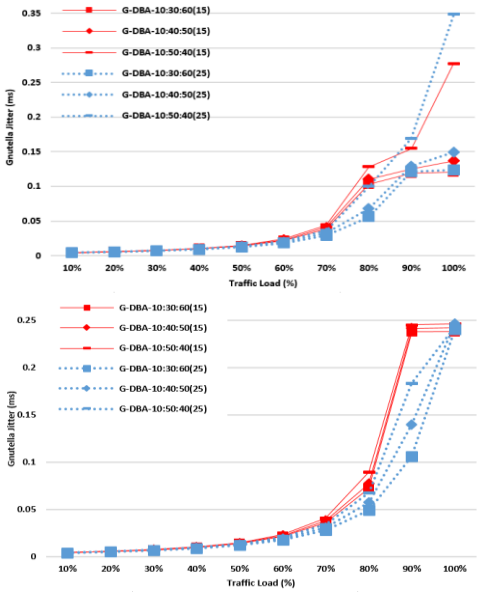
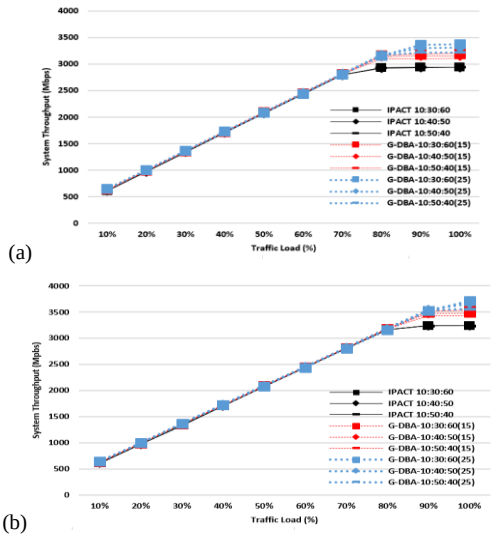


Fig. 10. Gnutella jitter with different traffic ratios for (a) 1.0ms cycle time (b) 1.5ms cycle time.

C. System Throughput

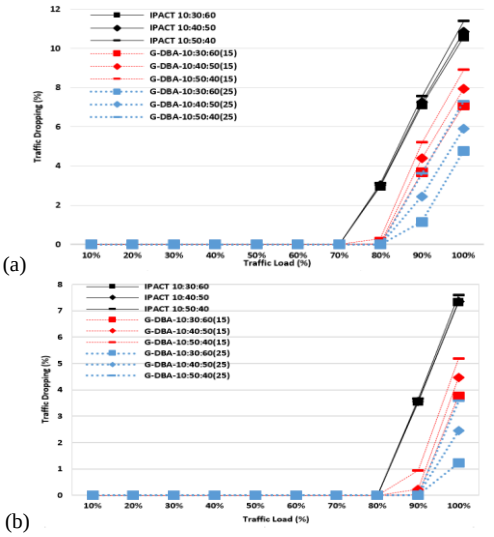
The system throughput is defined as the sum of the data rate that is transmitted to all terminals in the network which also includes the local (intra) traffic between the ONUs and users. Fig. 11 shows the system throughput in different cases with various traffic loads. The result clearly shows that the proposed architecture with Gnutella and redirect traffic through his higher than of IPACT traffic in both cycle times. For 1.0ms cycle time, we can improve our throughput to a maximum of 5.49% for Cases from 1 to 6 while for 1.5ms cycle time, the maximum improvement is 5%. The improvement is obvious because we are redirecting the intra traffic (15% and 25%) of BE traffic from one ONU to another. So when the number of intra traffic increases, the system throughput also increases. However, the system throughput performance at high traffic conditions (90% to 100%) are almost the same.



(a)

(b)

Fig. 11. System Throughput with different traffic ratios for (a) 1.0 ms cycle time (b) 1.5 ms cycle time.



(a)

(b)

Fig. 12. Traffic Dropping with different traffic ratios for (a) 1.0ms cycle time (b) 1.5ms cycle time.

D. Traffic Dropping

Cycle time and buffer size are the two main reasons for causing packet loss at ONUs. Increasing the cycle time reduces the packets lost because each ONUs has more time-slot to send its queues, however increasing cycle time also cause to a high packet delay. Increasing the buffer size reduces the packet loss but it will increase the queuing delay. Fig. 12 shows the improvement of traffic dropping for 1.0ms and 1.5ms cycle times. The simulation result shows the BE dropping has improved with the Gnutella traffic ratio (25%) of BE traffic as compared with (15%) of BE traffic. While comparing with both 1.0ms and 1.5ms cycle times, the result shows that the BE traffic dropping in 1.0ms cycle time is reduced up to 57% and in 1.5ms cycle time traffic dropping is reduced up to 77% in Case 6. From our observation, we concluded that the traffic loss occurs when the traffic load is beyond 70% at 1.0ms and 1.5ms cycle times, respectively. When the BE traffic ratio is higher, the BE traffic increases in all conditions. On the other hand, the packet loss has improved in the 1.5ms cycle time because the ONUs has more time to transmit the buffered packets.

IV. Conclusion

In this paper, we proposed a TWDM-PON architecture and integrated it with an SDN scheme to handle the Gnutella based applications. Our proposed G-DBA and SDN controller can handle and enhance the required bandwidth for Gnutella application as well as improve the transmission delay and success rate in intra PON. We used an extra wavelength to transmit the Gnutella traffic that assists faster file transfers in local PON, which increases the scalability, performance and guarantee the Quality of Services (QoS). Our proposed architecture can improve BE packet delays for up to 49%, EF packet up to 22%, AF packet up to 22%. Our throughput also went up to 5% and with dropping improvement up to 77% in Case 6 for 1.5ms cycle time. Likewise, the proposed architecture can be extended for multi-PONs to handle multiple Peer to Peer applications at the same time, such as Bit-Torrent, UTorrent, etc. Ultimately, the Network Function Virtualization (NFV) and Protection can be employed for all SD-OLT and SD-ONUs to provide a reliable system.

Conflict of Interest

This paper or its version does not submit to any journal.

This work was supported in part by the National Science Council of the Republic of China under grants MOST 107-2221-E-155-015.

Author Contributions

Ms. Ardian Rianto, Prof. Andrew Fernando Pakpahan and Prof. Andrew Tanny Liem studied the proposed the system architecture and Gnutella operations; Mr. Anish Sah is responsible for the simulation and data analysis; Prof. I-Shyan Hwang conducted the research and edited the paper; all authors had approved the final version.

References

[1] P2P Networking and P2P Software. [Online]. Available: https://www.lifewire.com p2p-networking-and-software-818019

[2] Y. Xu, X. Ma, and C. Wang, "Selective walk searching algorithm for Gnutella network," in *Proc. 4th IEEE Consumer Communications and Networking Conference*, 2007, pp. 746-750.

[3] V. Vishnumurthy and P. Francis, "A comparison of structured and unstructured P2P approaches to heterogeneous random peer selection," in *Proc. USENIX Annual Technical Conference.* USENIX Association: Santa Clara, CA, 2007, pp. 1-14.
[4] T. Androutsellis, *White Paper: A Survey of Peer-to-Peer File Sharing Technologies*, 2002.
[5] D. Spinellis, "A survey of peer-to-peer content distribution technologies," *ACM Computing Surveys*, vol. 36, no. 4, pp. 335-371, 2004.
[6] Gnutella Protocol Development. (2002). [Online]. Available: http://rfc-gnutella. sourceforge.net/src/rfc-0_6-draft.html
[7] Wikipedia, Gnutella. [Online]. Available: https://en.wikipedia.org/wiki/Gnutella
[8] P. Kisembe and W. Jeberson, "Future of peer-to-peer technology with the rise of cloud computing," *International Journal of Peer to Peer Networks*, vol. 8, no. 2/3, pp. 45-54, Aug. 2017.
[9] D. Stutzbach and R. Rejaie, "Characterizing the two-tier Gnutella topology," in *Proc. 2005 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems.* Banff, Alberta, Canada, 2005, pp. 402-403.
[10] M. Ripeanu, "Peer-to-peer architecture case study: Gnutella network," in *Proc. the First International Conference on Peer-to-Peer Computing*, 2001, pp. 99-100.
[11] D. Ilie and A. Popescu, "Statistical models for Gnutella signaling traffic," *Computer Networks*, vol. 51, no. 17, pp. 4816-4835, 2007.
[12] M. Portmann *et al*., "The cost of peer discovery and searching in the Gnutella peer-to-peer file sharing protocol," in *Proc. the 9th IEEE International Conference on Networks*, 2001, p. 263.
[13] L. Tsungnan, W. Hsinping, and W. Jianming. "Search performance analysis and robust search algorithm in unstructured peer-to-peer networks," in *Proc. IEEE International Symposium on Cluster Computing and the Grid*, 2004, pp. 346-354.
[14] C. J. Wu, K. H. Yang, and J .M. Ho, "AntSearch: An ant search algorithm in unstructured peer-to-peer networks," in *Proc. IEEE Symposium on Computers and Communications*, 2006, pp. 429-434.
[15] F. Ye, F. Zuo, and S. Zhang, *Routing Algorithm Based on Gnutella Model*, Springer Berlin Heidelberg, 2009, pp. 9-15.
[16] K. Althobaiti, S. J. Alotaibi, and H. Alqahtani, "Improving Gnutella query search algorithm with jumps," *International Journal for Information Security Research*, vol. 5, no. 4, pp. 600-607, Dec. 2015.
[17] S. Anurag and C. Rohrs. (2001). Ultrapeers: Another Step towards Gnutella Scalability. [Online]. Available: http://rfc-gnutella.sourceforge.net/Proposals/Ultrapeer/Ultrapeers.htm
[18] A. T. Liem, "P2P locality awareness architecture in Ethernet passive optical networks," in *Proc. International Conference on QiR*, 2013, pp. 111-115.
[19] Z. Wang *et al*., "MPCP design in prototyping optical network unit of Ethernet in the first mile," in *Proc. the Ninth International Conference on Communications Systems*, 2004, pp. 121-125.
[20] A. Dixit *et al*., "Novel DBA algorithm for energy efficiency in TWDM-PONs," in *Proc. 39th European Conference and Exhibition on Optical Communication*, 2013, pp. 1-3.
[21] X. Li and H. Yousefi'zadeh, "MPCP: Multi packet congestion-control protocol," *SIGCOMM Computer Communication*, vol. 39, no. 5, pp. 5-11, 2009.
[22] M. Li *et al*., "New dynamic bandwidth allocation algorithm for Ethernet PON," in *Proc. 8th International Conference on Electronic Measurement and Instruments*, 2007, pp. 224-227.
[23] B. Skubic *et al*., "Dynamic bandwidth allocation for long-reach PON overcoming performance degradation," *IEEE Communications Magazine*, vol. 48, no. 11, pp. 100-108, 2010.
[24] I. S. Hwang, A. Rianto, and A. F. Pakpahan, "Software-defined peer-to-peer file sharing architecture for TWDM-PON," in *Proc. 27th Wireless and Optical Communication Conference*, 2018, pp. 1-4.
[25] I. S. Hwang and A. T. Liem, "A hybrid scalable peer-to-peer IP-based multimedia services architecture," *IEEE/OSA Journal of Lightwave Technology*, vol. 31, no. 2, pp. 213-222, 2013.
[26] D. Nesset, "NG-PON2 technology and standards," *Journal of Lightwave Technology*, vol. 33, no. 5, pp. 1136-1143, 2015.
[27] M. Hajduczenia and H. J. A. D. Silva, "Next generation PON systems - Current status," in *Proc. 11th International Conference on Transparent Optical Networks*, 2009, pp. 1-8.
[28] Open Network Foundation, "Software-defined networking: The new norm for networks," ONF White Paper, 2012.
[29] D. Kreutz *et al*., "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14-76, 2015.
[30] Y. Zhao, B. Yan, and J. Zhang, "Software defined passive optical networks with energy-efficient control strategy," *Optik*, vol. 127, no. 23, pp. 11211-11219, 2016.
[31] A. F. Pakpahan, I. S. Hwang, and A. Nikoukar, "OLT energy savings via software-defined dynamic resource provisioning in TWDM-PONs," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 9, no. 11, pp. 1019-1029, 2017.
[32] P. Parol and M. Pawlowski, "Towards networks of the future: SDN paradigm introduction to PON networking for business applications," in *Proc. Federated Conference on Computer Science and Information Systems*, 2013, pp. 829-836.
[33] N. Cvijetic *et al*., "SDN and OpenFlow for dynamic flex-grid optical access and aggregation networks," *Journal of Lightwave Technology*, vol. 32, no. 4, pp. 864-870, 2014.
[34] Wikipedia, Gnutella2 Developer Network. [Online]. Available: http://g2.doxu.org/ index.php/Main_Page
[35] D. Ilie, "Gnutella network traffic measurements and characteristics," Blekinge Institute of Technology Licentiate Dissertation Series, 2006.
[36] G. Kramer, B. Mukherjee, and G. Pesavento, "IPACT a dynamic protocol for an Ethernet PON (EPON)," *IEEE Communications Magazine*, vol. 40, no. 2, pp. 74-80, 2002.
[37] G. Kramer, *Ethernet Passive Optical Network*, McGraw-Hill Professional, 2005.

Anish Sah received the B.C.A degree in computer science from the Purbanchal University of Nepal, in 2015, and the M.S degree from Yuan Ze University, Taoyuan City, Taiwan, in 2019. His research interests include high speed network, SDN, machine learning and IoT.

I-Shyan Hwang received B.S. and M.S. degrees in electrical engineering and electronic engineering from Chung-Yuan Christian University, Chung-Li, Taiwan, in 1982 and 1984, respectively, and M.S. and Ph.D. degrees in electrical and computer engineering from the State University of New York at Buffalo, NY, in 1991 and 1994, respectively. In Feb. 2007, he was promoted to full professor in the Department of Computer Science & Engineering at the Yuan Ze University, Chung-Li, Taiwan. His current research interests are fault-tolerant computing, high-speed networks, fixed mobile convergence, heterogeneous multimedia services over fiber optic networks, NGN, green computing and optical-network based infrastructure over cloud computing. He serves as a member of the editorial board for the Springer Photonic Network Communications Journal.

Ardian Rianto received a B.S. degree and an M.S. degree in computer science & engineering at Yuan Ze University, Taiwan in 2018. He is pursuing his Ph.D. degree now. His recent work focuses on peer-to-peer (P2P) applications in passive optical network.

Andrew Fernando Pakpahan received a B.S. degree in computer science from Universitas Advent Indonesia, Bandung, Indonesia and M.S. in informatics from Institut Teknologi Bandung, Indonesia, and received a Ph.D. degree in computer science & engineering at Yuan Ze University, Taiwan in 2018. His current research interests are in passive optical network, software-defined networking, and network function virtualization.

Andrew Tanny Liem received the B.S. degree from the Department of Computer Science, Adventist University of Indonesia, Bandung, Indonesia, in 2003, and the M.S. degree in 2006. He received the Ph.D. degree in computer science and engineering from the Yuan-Ze University, Taiwan in 2014. He is with Department of Computer Science at Klabat University, Manado, Indonesia. His recent work focuses on NGN and P2P over EPON and fault tolerance.