POTORI: A Passive Optical Top-of-Rack Interconnect Architecture for Data Centers

Yuxin Cheng, Matteo Fiorani, Rui Lin, Lena Wosinska, and Jiajia Chen

***Abstract*—Several optical interconnect architectures inside data centers (DCs) have been proposed to efficiently handle the rapidly growing traffic demand. However, not many works have tackled the interconnects at top-of-rack (ToR), which have a large impact on the performance of the data center networks (DCNs) and can introduce serious scalability limitations due to the high cost and power consumption. In this paper, we propose a passive optical ToR interconnect architecture (POTORI) to replace the conventional electronic packet switch (EPS) in the access tier of DCNs. In the data plane, POTORI relies on a passive optical coupler to interconnect the servers within the rack and the interfaces toward the aggregation/core tiers. The POTORI control plane is based on a centralized rack controller responsible for managing the communications among the servers in the rack. We propose a cycle-based medium access control (MAC) protocol to efficiently manage the exchange of control messages and the data transmission inside the rack. We also introduce and evaluate a dynamic bandwidth allocation (DBA) algorithm for POTORI, namely *Largest First* (*LF*). Extensive simulation results show that, with the use of fast tunable optical transceivers, POTORI and the proposed LF strategy are able to achieve an average packet delay below 10 μs under realistic DC traffic scenarios, which outperforms conventional EPSs. On the other hand, with slower tunable optical transceivers, a careful configuration of the network parameters (e.g., maximum cycle-time of the MAC protocol) is necessary to obtain a good network performance in terms of the average packet delay.**

***Index Terms*—Optical interconnect architectures, data center networks, medium access control (MAC), dynamic bandwidth allocation (DBA).**

I. Introduction

The growing popularity of modern Internet applications such as cloud computing, social networking and video streaming is leading to an enormous increase of data center (DC) traffic, including not only the north-south (client-server) traffic, but also the east-west (server-to-server) traffic exchanged within the DCs [1]. According to Cisco, the overall data center traffic will keep increasing at a compound annual growth rate (CAGR) of 25% up to the year 2019, reaching 10 Zettabytes per year [2]. Therefore, it is important to evolve the current data center network (DCN) infrastructure to support the continuously growing traffic demand.

Data center operators are addressing this problem by upgrading the transmission data rate and switching capacity of their network equipment. For example, Facebook has already deployed 10G Ethernet network interface cards (NICs) for all servers and Top-of-Rack (ToR) switches [3]. Optical fiber can be deployed in DCN to interconnect servers and switches in order to simplify cabling and avoid electromagnetic interference (EMI) [4]. Higher data rate and switching capacity (e.g., 40G, 100G) are also taken into consideration by the network operators in the future DC design [5]. However, it is hard to develop large electronic packet switches operating at high data rates, due to the bottleneck of I/O bandwidth and power budget of the chip [6]. As a consequence, a large amount of electronic switches need to be deployed to scale out the number of servers in the DC, which brings a serious scalability problem to the DCN in terms of cost and power consumption [7].

Optical interconnect architectures that are able to provide ultra-high transmission speed and switching capacity in a cost- and energy-efficient way, are considered to be a promising solution to address the limitations of electronic packet switches (EPSs) in DCs. By replacing EPSs with optical switches, the reduced power-demanding electrical-to-optical (E/O) and optical-to-electrical (O/E) conversion is expected to dramatically decrease the power consumption of data center networks [8]. Several optical interconnect architectures for DCs are proposed in literature in recent years, e.g., [13]—[16]. These architectures employ all-optical switches based on different topologies and technologies at aggregation/core layer, but rely on conventional EPSs at ToR to interconnect servers in the racks. However, the EPSs at ToR are responsible for a large amount of the overall DC traffic. For example, it is reported in [17] that in the DCs running extensive data exchange applications (e.g., MapReduce) around 80% of the total traffic is confined in the access tier. Moreover, the EPSs at ToR contribute to the majority of the overall DCN power consumption [21]. Therefore, efficient optical interconnect architectures for the access tier in DCs are required.

In our previous work (i.e., [8], [9], [21]), we proposed a concept of passive optical interconnect (POI) which uses mainly passive optical components for interconnection. The physical layer experiments [9] have shown that more than 500 ports can be supported in the passive optical interconnect at the capacity up to 5Tb/s. We also shown that the passive optical interconnect provides lower cost, lower energy consumption and higher scalability with respect to the conventional EPSs. Specifically, it has been demonstrated in [8] that the energy consumption per bit in the DCNs can be reduced by at least a factor of 7 by using passive optical interconnects at ToR compared to the ones using EPS. We also proposed a MAC protocol and a dynamic bandwidth allocation (DBA) algorithm, namely *Largest First* (*LF*), for achieving efficient bandwidth utilization when applying the passive optical interconnect at ToR [18]. This paper extends the work in [18] with a focus on passive optical ToR interconnect (POTORI) and introduces the following new contributions: (*i*) We illustrate how POTORI can be interconnected with other network architectures in the

The authors are with KTH Royal Institute of Technology, Department of Communication Systems, Electrum 229, SE- 164 40 Kista, Sweden (e-mail: {yuxinc, matteof, ruilin, wosinska, jiajiac}@kth.se).Corresponding author: Jiajia Chen.

aggregation/core tier to build large DCNs, where both the data plane and control plane are considered; (*ii*) We perform an extensive performance comparison among different DBA algorithms for POTORI; (*iii*) We study the impact of different network configuration parameters (e.g., tuning time of the optical transceivers, duration of the cycle time in the MAC protocol, etc.) on the performance of POTORI; and (*iv*) We compare the performance of POTORI with a conventional EPS in terms of the average packet delays and packet loss probability. The results show that using our proposed LF DBA algorithms along with ultra-fast tunable transceivers, POTORI can outperform the conventional EPS.

The rest of the paper is organized as follows. We present the related works on optical interconnect architectures for DC in Section II. In Section III, we illustrate the POTORI architecture, including both data plane and control plane. The proposed centralized MAC for POTORI is elaborated in Section IV, and in Section V we introduce and analyze the proposed DBA algorithm. The simulation results of POTORI and a conventional EPS are presented and discussed in Section VI. We conclude the paper in Section VII.

II. Related Works

Several optical interconnect architectures for DCN have been proposed in the literature. The c-through [10] and HOS [11] are two examples of hybrid electronic/optical interconnection solutions. In these hybrid interconnect architectures, optical circuit switches (OCS) are employed to provide high capacity for transmitting long-lived and bandwidth-consuming traffic flows (e.g., elephant flows) due to the long reconfiguration time of OCS, and EPS is used to transmit short-lived traffic flows that do not need large bandwidth (e.g., mice flows). These solutions require pre-knowledge or classification of traffic pattern in order to distinguish large and small flows and properly configure the OCS, which is challenging for DC operators.

On the other hand, there are some all-optical architecture solutions proposed recently. In [12], the authors demonstrated a new optical circuit switching (OCS) based architecture for DCN, which is based on a single comb-driven MEMS mirror and is able to achieve a switching time of 20 μs. However, such fast switching might still create a substantial delay in case of a small amount of data to be transmitted, making it not suitable to be employed at ToR where small bursts of intra-rack traffic need to be handled. The authors in [13] proposed a flat data center network architecture with fast flow control. Each ToR switch is connected to one intra-cluster optical switch as well as one inter-cluster optical switch. All the traffic is switched by optical switches according to the flow control mechanism, which is based on the packet header processing on each electronic ToR switch. OPMDC [14] is a three-tier architecture, where each tier is a set of reconfigurable optical add-drop multiplexers (ROADM) connected in ring topology, and the ROADM rings in the lower tier are connected to a ROADM in the upper tier. At the access tier each ToR switch is connected to a single ROADM. Space division multiplexing (SDM) is also considered in all-optical DCN solutions, e.g., [15][16] for improving capacity. In [15], four architectures based on SDM are proposed, and it is shown that these architectures are suitable to apply in different DCs depending on their size and work load. The authors in [16] reported an optical data center architecture based on multidimensional switching nodes connected in ring topology. These switching nodes are able to switch in space, wavelength and time domains, supporting the connections of different granularities. The ring topology reduces the number of physical links, simplifying the cabling management. Nevertheless, all these aforementioned architectures rely on optical switching only in the aggregation/core tires, while they are based on conventional electronic ToR switches in the access tier to interconnect the serves within the same rack. In [20], the authors proposed and demonstrated software-defined ubiquitous data center optical interconnection (SUDOI), which also considers optical switch at ToR. However, the main focus of SUDOI is on the control plane, where a service-aware schedule scheme is introduced to enable cross-stratum optimization of application and optical network stratum resources while enhancing multiple-layer resource integration. The concrete design of optical interconnects within DCs is not provided.

In [21] and [22] we proposed several passive optical ToR interconnect architectures for the access tier in DCs. These architectures use passive optical components (i.e., arrayed waveguide grating (AWG) and/or optical couplers) to interconnect the servers in the rack. It has been demonstrated that the passive optical components offer cost and power saving as well as high reliability. While in [21] and [22] we focused on the data plane architecture, in [18] we proposed a MAC protocol and novel DBA algorithms to achieve efficient bandwidth utilization in POTORI. The current paper extends our previous work by covering both data and control plane, and provides a complete design of POTORI architecture along with a detailed analysis of the network performance and an extensive comparison with the conventional EPS solutions.

III. POTORI: Passive Optical Top-of-Rack Interconnects

Fig. 1 illustrates the POTORI architecture, including both data plane and control plane. Each server is equipped with an optical network interface (ONI), which consists of two optical transceivers. The first one is a tunable transceiver connected to the POTORI data plane. The second one is a grey small form factor pluggable (SFP) transceiver connected to the POTORI control plane. In the following subsections, we elaborate the POTORI data and control planes, mapping POTORI to the use case of DCNs.

Data Plane

The POTORI data plane was proposed and introduced in our previous work [22]. The key component of POTORI data plane is an (N+1)×(N+1) passive coupler that acts as the switching fabric to interconnect all the servers in the rack. We define N as the number of servers in the rack. In each ONI, the tunable transceiver is composed of a wavelength tunable transmitter (WTT) for transmitting data, and a wavelength tunable filter (WTF) as well as a receiver for receiving data. The WTT and WTF are connected to one input port and one output port of the coupler, respectively. One additional pair of input and output ports of the coupler are connected to a wavelength selective switch (WSS), which forwards and receives the traffic to/from the aggregation and core tiers.

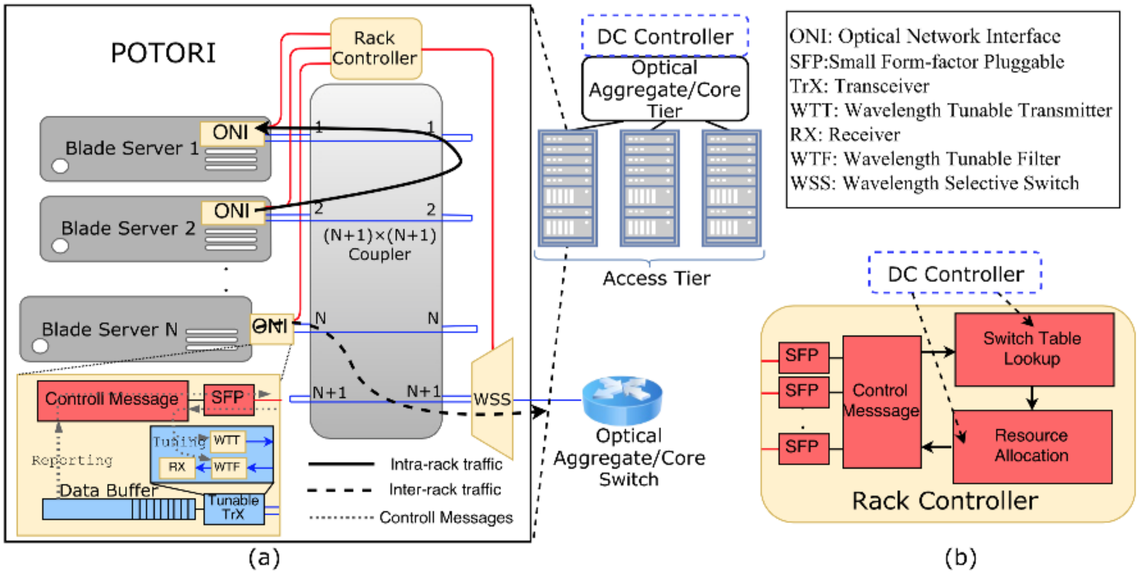

Fig. 1: (a) POTORI Architecture in Data Center, (b) Rack Controller

It can be seen that the POTORI's data plane is passive, except the WSS that is needed to dynamically filter out spectrum for the inter-rack communications. Actually, WSS can be replaced by the passive wavelength filter, in which a fixed configuration of the spectrum for intra- and inter-rack communications may result in low flexibility of resource usage. Due to the key component — coupler, the data transmission in POTORI follows the broadcast-and-select scheme. The traffic transmitted by one server is broadcast and received by all the other servers and the WSS. The destination server (or the WSS) then selects the traffic destined to it, while discarding the other traffic. In this way, the servers are able to send/receive traffic to/from each other in the rack (e.g., Server 2 sends traffic to Server 1 in Fig.1). The WSS receives and drops the intra-rack traffic while forwarding the inter-rack traffic to/from the upper tier (e.g., Server N sends traffic to the aggregation/core tier in Fig.1).

Control Plane

In order to successfully transmit data, both ONIs at source server and destination server need to be tuned to the same wavelength. To avoid data collision in the coupler, concurrent communications inside the rack can be carried on the different wavelengths. It calls for a proper control plane design to efficiently schedule resources in both spectrum and time domains for managing the intra-rack and inter-rack communications.

The proposed centralized control entity for POTORI, namely rack controller, is shown in Fig. 1. The rack controller exchanges control information with the ONIs using dedicated control links. The servers report the relevant information (e.g., buffer size) to the rack controller and tune the transceivers according to the feedback from the rack controller. The POTORI MAC protocol defines the exchanging procedure and the format of the control messages between the servers and the rack controller, which will be elaborated in Section IV. On the other hand, the rack controller collects the necessary traffic information from each server and creates the traffic matrix. Then it runs a DBA algorithm, determining the wavelength and time slots assigned for all the servers in the rack. Finally, it generates the control messages that include the outcome of the DBA and sends them to each server.

Application of POTORI in Data Center Networks

The POTORI architecture can be interconnected with any solution for the aggregation/core tiers to build large DCNs. In the data plane proper interfaces are needed to interconnect the POTORI with the aggregation/core switches. These interfaces can employ O/E conversion for connection to the conventional EPS in aggregation/core tier or they can be optical (e.g., to directly connect the POTORI to an optical core switch and realize an all-optical DCN [8]). In the latter case, a strategy for the joint allocation of the optical resources in the access and aggregation/core tires needs to be developed.

In the control plane the rack controller can be connected to a higher layer DC controller in a hierarchical control architecture (see Fig. 1(b)). In this way the DC operator can employ a single control infrastructure to manage all the resources in the DC. Depending on how the DC controller interacts with the rack controller, two different modes of operation can be defined, namely fixed-mode and programmable-mode. In the fixed-mode the DC controller is not able to influence the resource allocation inside the rack. The rack controller performs layer 2 functions, such as switch table lookup, and computes the resource allocation according to a deployed DBA algorithm. On the other hand, in the programmable-mode (see Fig. 1(b)) the DC controller can influence the resource allocation inside the rack, e.g., by changing the employed DBA algorithm dynamically. A possible way to realize a control plane operating in programmable-mode is to equip the rack controller with a configurable switch table (e.g., an OpenFlow [24] switch table) and a configurable resource allocation module (see Fig. 1(b)). Using a software defined networking (SDN) [23] DC controller is then able to dynamically change the flow rules and the DBA algorithm employed by the rack controller. In this paper, we consider only the control plane in fixed-mode, and leave the programmable-mode for the future work.

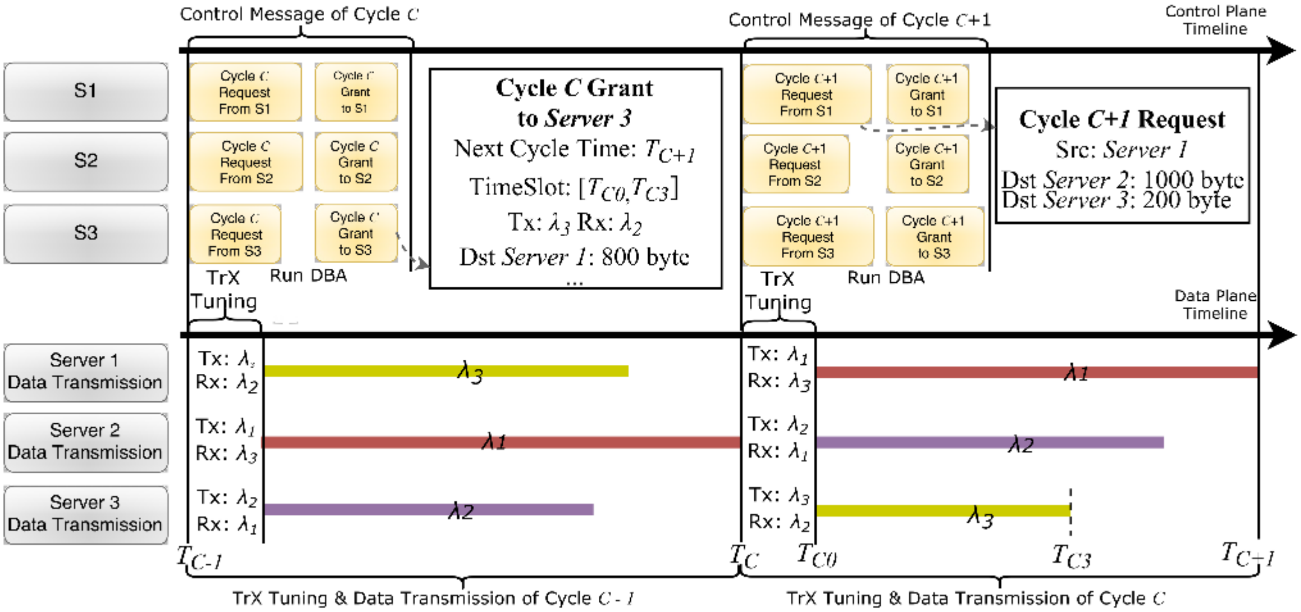

Fig. 2: The MAC Protocol and the Request-Grant Messages of POTORI

IV. MAC Protocol

Due to the broadcast property of the coupler, POTORI requires a proper MAC protocol to efficiently coordinate the data transmissions among the servers in a rack. The existing MAC protocols can be classified as distributed and centralized. In distributed MAC protocols, every server determines its own resources for data transmission, based on the direct exchange of control information with the other servers in the network and without the involvement of the centralized controller. There are some typical examples of distributed MAC protocols, such as the carrier sense multiple access with collision detection (CSMA/CD) [25] which has been widely employed in the old version of Ethernet (10 Mb/s and 100 Mb/s), and the high-efficient distributed access (HEAD) protocol which was proposed for the optical interconnect architecture POXN in [26]. However, the control overhead in the distributed MAC protocol should be taken into account. Given the large amount of short lived traffic flows within a rack in data centers, the control overhead brought by the distributed MAC protocol might be significant and the performance of the network (packet delay, packet drop ratio, etc.) will decrease. As a consequence, the distributed MAC protocols might not be good candidates for POTORI.

In contrary, a centralized controller is able to manage the exchange of control information and coordinate the data transmission among all the servers in the network. Typical examples are the IEEE 802.11 protocols [27], which are used in Wi-Fi networks, the multipoint control protocol (MPCP) [28] which is used in Ethernet passive optical networks (EPONs), and the time division multiple access (TDMA) Ethernet proposed in [29]. The TDMA Ethernet seems to be a good candidate for POTORI, as it allows one server at a time to use the entire available bandwidth for transmission in order to achieve low latency and low packet drop in the network. However, applying TDMA Ethernet to POTORI would result in a single server transmitting at a time over one wavelength and suspending the communications among the remaining servers. POTORI is able to support concurrent communications using wavelength division multiplexing (WDM), where multiple wavelengths are utilized to set up connections for different server pairs. The MPCP is another choice for POTORI, due to the similarity of the data plane in PON and POTORI. However, POTORI requires multipoint-to-multipoint communications where all the servers should be able to communicate with each other, while MPCP supports only multipoint-to-point communications, and thus it cannot be directly applied to POTORI. In addition, IEEE 802.11 standards do not support WDM, so that the control schemes utilized in Wi-Fi networks are not applicable to POTORI. In this regard, we propose a new centralized MAC protocol tailored for the POTORI architecture.

The proposed MAC protocol is shown in Fig. 2 and follows a Request-Grant approach. The time is divided in cycles. At the beginning of each cycle, each server sends a Request message to the rack controller. The Request message contains the information about the packets currently buffered at the server. After receiving the Request messages from all the servers, the rack controller is able to generate a traffic matrix and run the DBA algorithm to calculate the allocation of wavelengths and transmission time for all the servers. Afterwards, the controller informs the servers about the outcome of the resource allocation using Grant messages, which contain all the necessary information (wavelength, cycle time, etc.) for configuring the ONIs of the servers. At the beginning of the next cycle, each server will tune the WTT and WTF to the assigned wavelengths for transmitting and receiving data according to the information in the Grant message. For example, in Fig. 2, at the time T_{C-1}, all three servers (S1, S2, S3) report their buffer information to the rack controller through Request Messages for Cycle *C*, and the rack controller then responds with the Grant Messages in the same cycle. Meanwhile, all three servers tune their WTTs and WTFs, according to the Grant Message received during the previous cycle (i.e., Cycle *C-1*) and start transmitting and receiving traffic. At the time T_C (specified in the Grant Messages received during Cycle *C*), all the servers tune the WTTs and WTFs accordingly. In the example shown in Fig. 2, Server 3 tunes the transmitter to λ_3 and the receiver to λ_2, and Server 1 tunes the transmitter to λ_1 and the receiver to λ_3 for Cycle *C*. Consequently, the traffic from Server 3 to Server 1 can be successfully transmitted in this cycle, which lasts from T_{C0}, to T_{C3}, according to the Grant message. Similarly, the Request and Grant Messages for Cycle *C+1* are exchanged in parallel with the data transmission of Cycle *C*. Thanks to the dedicated connection between each server and the rack controller, collisions among the Request and Grant Messages can be eliminated.

Note that each server only transmits the granted data according to the Grant message, which might be only a portion of the ones reported in the Request message. It is worth to mention that the granted traffic, which will be transmitted in the next cycle, is not reported in the next Report Message. For example, in Fig. 2 Server 3 receives the Grant Message of Cycle *C*, allowing to send 800 bytes to Server 1. At T_C, Server 3 subtracts the 800-byte traffic and reports the remaining data with the destination of Server 1 in the Request Message of Cycle *C+1*.

We further illustrate the structure of the Request and Grant Messages in the following sections.

A. *Request Message*

Fig. 2 shows an example of the Request message. The first field of the Request message contains the current time cycle identifier (e.g., Cycle *C+1* in Fig. 2), which is used by the rack controller to identify whether the received control messages are outdated or not. If a Request Message is not synchronized with the cycle identifier at the rack controller, it is discarded by the rack controller. The second field of the Request Message contains MAC address of the source server, i.e., the server that generates the Request message (e.g., Server 1 in Fig. 2). Besides, the request message should also contain in total *N* fields for all the possible destination MAC addresses (*N-1* for the other servers in the rack and one for the interface toward the aggregation/core tier), along with the corresponding number of buffered bytes, i.e., the bytes to be transmitted at the source server.

The Request Message can be encapsulated in an Ethernet frame. The length in bytes of the Request message (L_R) can be calculated as:

$$L_R = L_{CH} + L_{SRCMAC} + N \times (L_{DSTMACi} + L_{Si}) \quad (1)$$

where L_{CH} is the length of the first field in the time cycle, L_{SRCMAC} and $L_{DSTMACi}$ are the length of MAC address of the source and the i_{th} destination (6 bytes) server, and L_{Si} is the length of buffed packets size for i_{th} destination. If we assume L_{CH} = 8 bytes, L_{Si} = 4 bytes, a Request message as an Ethernet frame with the maximum size (1518 byte) can support up to 150 servers in a rack, which is sufficient for the access tier in DCs.

B. *Grant Message*

An example of the Grant message is shown in Fig. 2. Similar to the Request Message, the first field of Grant Message contains the current cycle identifier (e.g., Cycle *C* in Fig. 2). According to this field, the servers that newly join the network or lose synchronization to the rack controller can update their cycle identifier. The second field contains the timestamp that indicates the end of the cycle (e.g., T_{C+1} in Fig. 2). The third field contains the destination MAC address of the Grant Message, i.e. the server which the Grant message is destined to (e.g., Server 3 in Fig. 2.). The following three fields contain: (*i*) a time slot with the starting timestamp (e.g., T_{C0} in Fig. 2) and ending timestamp (e.g., T_{C3} in Fig. 2) for transmission; (*ii*) the assigned wavelengths for transmitter (e.g., λ_3 in Fig. 2) and receiver (e.g., λ_2 in Fig. 2) during the timeslot given in (*i*); (*iii*) the destination MAC address (e.g., Server 1 in Fig. 2) as well as the granted size (e.g., 800 bytes in Fig. 2) for the transmission during the time slot given in (*i*). The time slot can last either the entire time cycle or a part of the time cycle. Note that we define an extra parameter T_M as the maximum transmission time for each cycle. Each server cannot transmit data longer than T_M in each cycle, i.e., in Fig. 2 T_{C+1} - T_C should be less or equal to T_M. The value of T_M definitely affects the network performance, which will be discussed in Section VI.

The length of a Grant message L_G can be calculated as:

$$L_G = L_{CH} + L_{SRCMAC} + L_{NCT} + 2 \times L_{WI} + L_{DSTMAC} + L_s \quad (2)$$

where L_{NCT} is the length of timestamp, L_{TS} is the length of the starting/ending timestamp for data transmission, and L_{WI} is the length of the wavelength identifier. The remaining symbols are the same as the ones for the Request Message. If we would consider a timestamp of 8 bytes and a wavelength identifier of 1 byte, the length of a Grant message would be 52 bytes which is small enough to be encapsulated into one Ethernet frame.

V. Dynamic Bandwidth Allocation Algorithms

A traffic demand matrix can be built by the rack controller after receiving the Request Messages from all servers and uplink interfaces. An example is shown in Fig. 3. The rows and columns of the matrix indicate the input port (source) and output port (destination), respectively, and the matrix element represents the amount of traffic (in bytes) that needs to be transmitted from the source to the destination. The DBA algorithm should find a solution for assigning available wavelengths to the different traffic demands without any collision in the data transmission. A collision occurs when different traffic demands are assigned to the same wavelength at the same time. To avoid collisions, at most one traffic demand can be assigned to an available wavelength, i.e., each row and each column in the matrix should be associated to exactly one wavelength. The right side of Fig. 3 gives a feasible solution of the wavelength assignment without any collision. The wavelengths assigned for serving the different traffic demands are distinguished by colors. The rack controller forms the Grant Messages according to the DBA solution, indicating the wavelengths for transmitting and receiving at every server.

The problem described above is similar to the classical switch scheduling problem. A conventional electronic switch buffers incoming data traffic at input queues, and then forwards it to the output ports. The traditional crossbar switch fabric allows multiple input queues to transmit data traffic to different output ports simultaneously. Many scheduling solutions have been proposed over decades, aiming to find

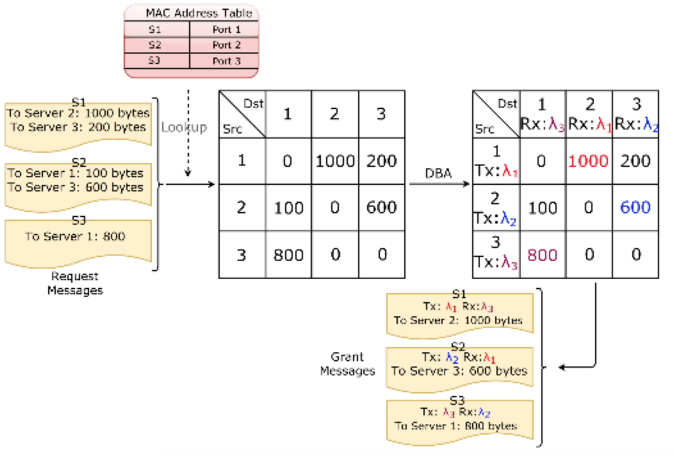

Fig. 3: Traffic Demand Matrix.

matches between input port and output port to achieve high throughput and low packet delay. Some solutions are based on matrix decomposition. For example, paper [30] uses Birkhoff von Neumann (BvN) algorithm to find optimal scheduling solutions to configure the circuit switches in DC. However, these solutions are not suitable for POTORI, since the high time complexity of matrix decomposition algorithms makes it only feasible for the scenario in [30], where the traffic demand for a long time period is known before running the algorithm. On the other hand, there are some other solutions having lower time complexity, such as iSLIP [19], which is one of the most widely used in electronic switches. However, iSLIP is not designed to support WDM. Thus we adapt the iSLIP algorithm to POTORI as a benchmark DBA algorithm to be compared with ourproposed novel DBA algorithm, referred to as *Largest First* (*LF*). We define N and W as the size of the matrix and the number of available wavelengths, respectively.

A. Benchmark: iSLIP Adapted to POTORI

In this paper we adapt iSLIP to be used in POTORI. The iSLIP algorithm is an improved Round-Robin Matching (RRM) algorithm. Each input and output port of a crossbar switch fabric is associated a Round-Robin (RR) scheduler. The detailed procedure of iSLIP algorithms can be found in [19]. Within N iterations, the iSLIP algorithm is able to find up to N matches between input and output from a traffic demand matrix. Given a traffic demand matrix, suppose that iSLIP algorithm finds $N*$ matches, where $N* \leq N$. In the adapted iSLIP for POTORI, these found matches needs to be assigned different wavelengths. If $W \geq N*$, then it is possible to assign every match with a unique wavelength. When $W < N*$, we randomly pick W matches from iSLIP's result and assign them with different wavelengths. In original iSLIP algorithm, whenever an input-output match is found, the corresponding scheduler of this match is updated. The adapted iSLIP for POTORI updates the schedulers if and only if a match is assigned a wavelength.

The iSLIP algorithm is easy to implement in the hardware. It achieves 100% throughput for uniformly distributed Bernoulli arrivals, but may not be efficient for bursty arrival traffic patterns [19] [31], which is often more suitable to model the real traffic pattern in DCs [32].

Algorithm 1 Largest First Algorithm

```
 1: Input: M; W; const R
 2: %Input: TDM M, wavelength list W, transceiver data rate R
 3: $tX \leftarrow [None, None...]$; $txTime \leftarrow [0, 0]$
 4: $rX \leftarrow [None, None...]$; $rxTime \leftarrow [0, 0]$
 5: List T ← $\mathbf{M}.sort()$
 6: repeat
 7:     $D \leftarrow \mathbf{T}[0]$
 8:     if $D.tX$ is None and $D.rX$ is None then
 9:         $D.assigned \leftarrow$ True
10:         $tX[D.src] \leftarrow [\mathbf{W}[0] : [0, D.size/R]]$
11:         $rX[D.dst] \leftarrow [\mathbf{W}[0] : [0, D.size/R]]$
12:         $txTime[D.src] \leftarrow D.size/R$
13:         $rxTime[D.dst] \leftarrow D.size/R$
14:         delete $\mathbf{W}[0]$
15:     delete $\mathbf{T}[0]$
16: until T or W is Empty
17: return $tX, rX$
```

Fig. 4: Largest First Algorithm

B. Largest First

The Largest First (LF) is a greedy heuristic algorithm. It prioritizes the largest element in the traffic demand matrix, i.e., larger amount of traffic demand has higher probability to be assigned with a wavelength. First, the matrix elements are sorted in a descending order into a one-dimensional array (Line 5 in Fig. 4). Then, starting from the first element in the array, a traffic demand is assigned with a wavelength if and only if neither the transmitter nor the receiver associated to this demand have already been assigned another wavelength in the current cycle (Line 7-9 in Fig. 4). The corresponding information such as wavelength, source, destination and transmission time is used to generate the Grant Message (Line 10-13 in Fig. 4). If one of the transmitter or receiver of this demand is assigned another wavelength, the demand is not served and left for the next cycle (Line 15 in Fig. 4). The LF algorithm stops when all the available wavelengths are assigned, or the last traffic demand in the array is served (Line 16 in Fig. 4).

VI. Performance Evaluation

In this section, we evaluate the performance of POTORI and compare it with conventional electronic ToR packet switch (EPS) in terms of average packet delay and packet drop ratio. To be more specific, the packet delay consists of queuing time at the source node (servers and uplink interfaces), transmission time and propagation time. In POTORI, the ONI at servers drops packet when the buffer is full. Moreover, we examine the impact of different system configurations (e.g., selected DBA algorithms, tuning time of the transceivers, etc.) on the performance for POTORI. We build a customized discrete-event-driven simulator implemented in Java for the performance evaluation.

A. Traffic Model

The traffic model used in simulations is derived from [17][32]. Each server generates 10^6 packets where the packet inter-arrival time follows a lognormal distribution. The size of packets follows a bimodal distribution (i.e. most of packets are with size of either 64-100 bytes or around 1500 bytes), which is shown in Fig. 5(c). The data rate (R) per server is set to 10 Gb/s and we assume that the buffer size on the

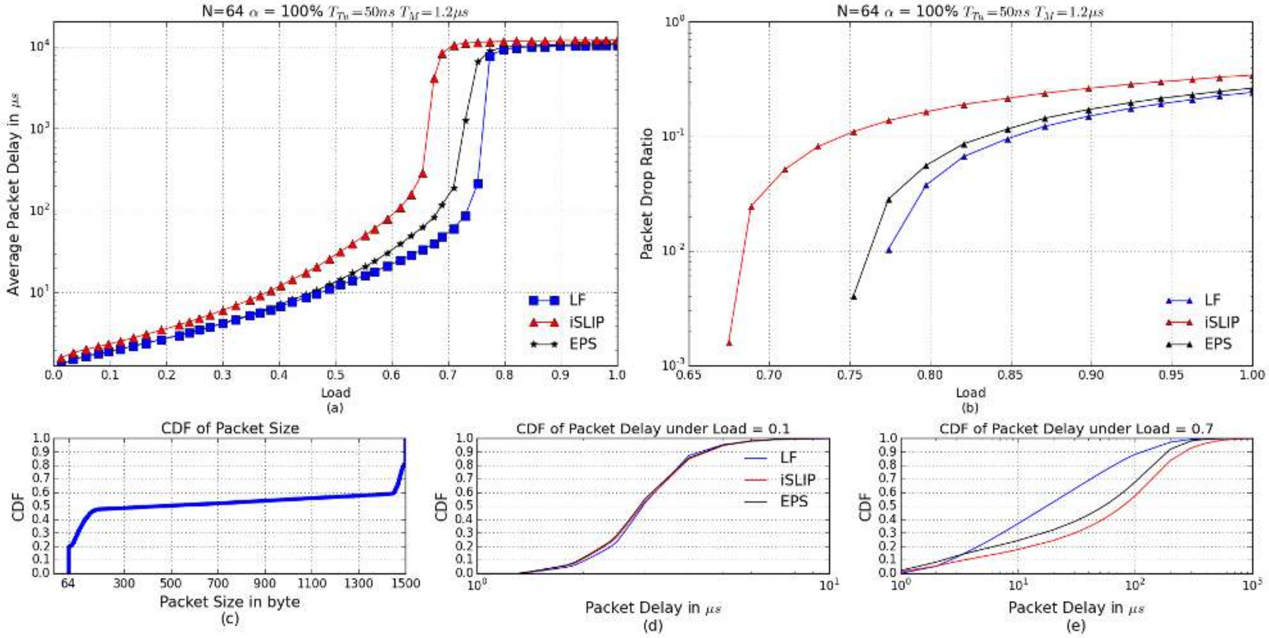

Fig. 5: (a) average packet delay; (b) packet drop ratio; (c) CDF of packet size for simulations (d) CDF of packet delay under load = 0.1; (e) CDF of packet delay under load = 0.7.

TABLE I: Table of Notations

N	Number of Servers in a rack
R	Data Rate of the Tunable Transceiver
O_R	Oversubscription Rate
α	The ratio between the number of available wavelengths and the sum of the number of servers and uplink transceivers
T_{Tu}	Tuning Time of the Transceiver
T_M	Maximum Transmission Time of Cycle
T_P	Transmission Time of an Ethernet Frame with size of 1518 bytes

ONI is 10 Mbytes. The propagation delay is set to 50 ns, which corresponds to 10 m fiber for interconnect within the rack. The network oversubscription rate O_R is set to 1 : 4 (i.e., if we consider 64 servers in a rack, there are 16 tunable transceivers for communication with the aggregation/core tiers, and the coupler with 64+16 = 80 pairs of input and output ports interconnecting all servers and uplink transceivers). Servers and uplink ports generate packets with random destinations. We assume that 80% of the traffic generated by servers is intra-rack traffic. The destination of the intra-rack traffic is uniformly distributed to all the other servers in the rack. The remaining 20% of traffic is inter-rack, whose destination is uniformly distributed among the uplink interface transceivers. Meanwhile, each uplink interface transceiver generates packets with destination uniformly distributed to all servers, representing the traffic from aggregation/core layer to the rack. We define N as the number of servers in a rack, and α as the ratio between the number of available wavelengths for a rack and N(1+O_R) (i.e., the sum of the number of servers and uplink transceivers), reflecting the sufficiency of the wavelengths. In addition, the tuning time of the tunable transceivers is defined as T_{Tu}, and the maximum transmission time of a cycle is defined as T_M. Finally, Table I summarizes all the notations.

B. POTORI v.s. EPS

In this subsection, we compare the performance of POTORI using different DBA algorithms with the conventional EPS. We set the line rate to10Gb/s for both POTORI and EPS. The tuning time of the tunable transceiver for POTORI is 50 ns (i.e., T_{Tu} = 50 ns) [29], and the maximum transmission time T_M for each cycle is set to 1.2 μs, allowing the server to transmit at most one packet with the maximum size of 1518 bytes. Fig. 5 (a) and (b) show the average packet delay and packet drop ratio for a rack with 64 servers (i.e., N = 64) and α = 100%, which means the total number of available wavelengths is $N\times(1+O_R)$ = 80).

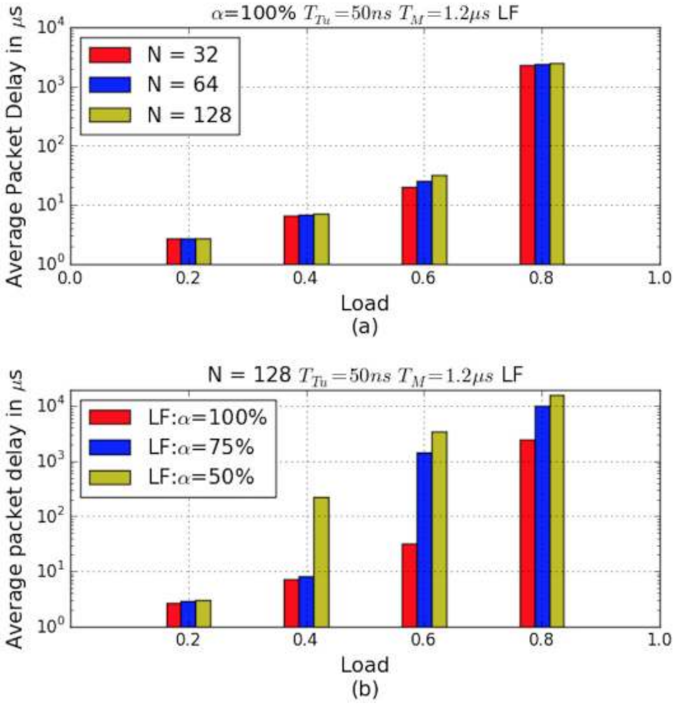

Fig. 6: Average packet delay with (a) different N; (b) different α.

It can be seen in Fig. 5(a) that when load is lower than 0.5, POTORI with the proposed LF DBA algorithm can achieve a

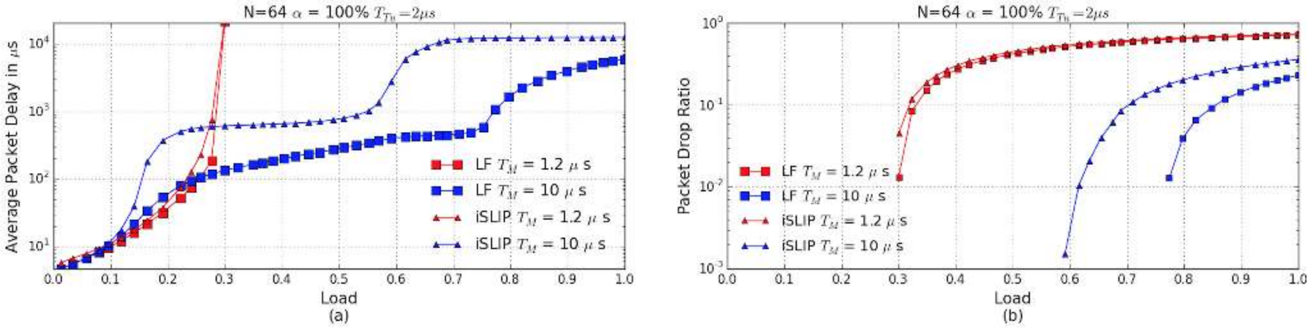

Fig. 7: Performance with T_{Tu} = 2 μs and different T_M: (a) average packet delay; (b) packet drop ratio.

packet delay lower than 10 μs, which performs similar as EPS. When the load is increased to 0.7, POTORI with LF is able to introduce up to 50% lower delays compared to EPS, thanks to the LF's feature of prioritizing the large traffic demand. Moreover, it performs slightly better (around 2% difference) than EPS in terms of the packet drop ratio. On the other hand, POTORI with the iSLIP DBA algorithms, has the worst performance. When the load is lower than 0.5, the average packet delay when employing iSLIP is double as high as that with LF. In addition, iSLIP shows highest packet drop ratio, which is greater than 10% when load is larger than 0.75.

In Figs. 5(d) and 5(e), we show the cumulative distribution function (CDF) of the packet delay at load = 0.1 and load = 0.7, which represent the cases with low load and high load, respectively. Under the load of 0.1, the difference between POTORI and EPS is negligible, and POTORI is able to achieve packet delay less than 10 μs for almost all packets (>99.99%) . Under the load of 0.7, there are 40% of the packets that are transmitted to the destination within 10 μs by POTORI with LF, and 80% of the packets have a delay lower than 100 μs, which outperforms POTORI with iSLIP and EPS.

C. Impact of Network Configuration

In this subsection, we investigate the impact of network configuration on performance and present the average packet delay as a function of *N* and α. The T_{Tu} and T_M are set the same in the previous subsection (i.e., T_{Tu} = 50ns and T_M = 1.2 μs). Because the proposed LF obviously outperforms iSLIP, we choose LF as the DBA algorithm for POTORI.

Fig. 6(a) shows the average packet delay of POTORI with different number of servers (i.e., N = 32, 64, and 128) in the rack with α = 100% (corresponding to 40, 80, and 160 available wavelengths, respectively). Under the low load (i.e., load = 0.2) and the heavy load (i.e., load = 0.8), the difference in the average packet delay is very small. It is because under the low load, the wavelength resources are sufficient and most of the traffic demands can be transmitted in one cycle regardless of *N*, while under high load the system gets statured, always resulting in high delay. For the medium load (i.e., load = 0.4 and 0.6), the packet delay performance degrades with the increase of the number of servers. This is due to the fact that the number of available wavelengths (which is equivalent to the maximum number of assigned traffic demands in each cycle) increases linearly with the number of servers, but the total number of traffic demands increases quadratically.

Fig. 6(b) shows the average packet delay with different values of α. We consider 128 servers per rack (i.e., *N* = 128), and test α =100% (160 wavelengths), 75% (120 wavelengths), and 50% (80 wavelengths). The average packet delays are almost the same for all the considered values of α under the low load (i.e., load = 0.2), indicating that in this condition using less wavelengths in POTORI under the low load still maintains good packet delay performance (less than 10 μs). However, the packet delay with α = 50% increases dramatically under the higher load, while with α = 75%, the performance decreases significantly at load = 0.6.

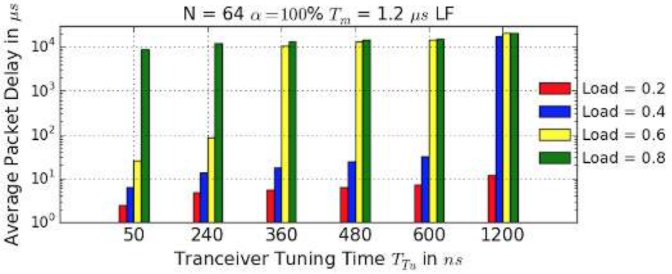

Fig. 8: Average packet delay with different T_{Tu}.

D. Impact of Transceivers' Tuning Time T_{Tu}

In the previous subsections, we considered a tunable transceiver with an ultra-fast tuning time T_{Tu} = 50 ns and a maximum transmission time of T_M = 1.2 μs for each cycle. Note that with T_M = 1.2 μs, packet-level switching granularity is achieved by POTORI, since in each cycle there is at most one packet transmitted by a server. However, with the larger tuning time it may become challenging to efficiently realize the packet-level switching granularity. Thus, we relax the constraint of transceivers tuning time to 2 μs, which has been reported by the commercially available products [34]. With T_{Tu} = 2 μs, the performance of POTORI in terms of average packet delay and packet drop ratio with different DBA algorithms are shown in Fig 7. If keeping T_M = 1.2 μs, the packet delay of both DBA algorithms increase tremendously and the system has a high packet drop ratio, due to the increased tuning overhead. However, the performance can be much improved by increasing T_M. With T_M = 10 μs, the packet delay for POTORI with LF DBA algorithm employed can still be maintained below 100 μs when the load is lower than 0.3, and below 400 μs under the load up to 0.7. When employing iSLIP, on the other hand, POTORI performs worse, as the packet delay is around 600 μs under the medium load (load from 0.3 to 0.6). In addition, LF achieves lower packet drop ratio compared to iSLIP. Nevertheless, for both the LF

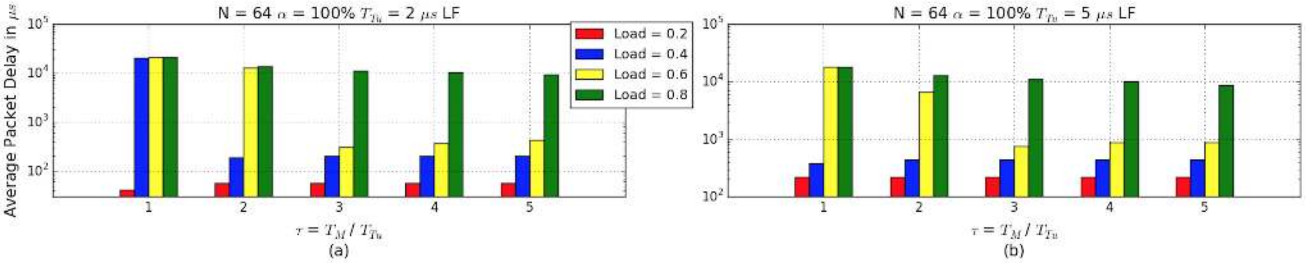

Fig. 9: Average packet delay with different ratio between T_M and T_{Tu}: (a) T_{Tu} = 2 μs; (b) T_{Tu} = 5 μs.

and iSLIP setting T_M = 10 μs outperforms the case with T_M = 1.2 μs. The reason for this is that with larger T_M, more packets can be sent by servers within one cycle, which reduces the tuning overhead.

One important question is "how to set the proper value of T_M for POTORI". Obviously, the proper value of T_M highly depends on the tuning time T_{Tu} of the transceiver. POTORI is able to achieve the packet-level switching granularity with the fast tuning transceivers. Fig. 8 shows that a moderate packet delay performance (i.e., < 100 μs) when the load is lower than 0.6 can be achieved at the tuning times equal to 50 ns and 240 ns (corresponding to 4% and 20% of the maximum transmission time of one packet). The packet delay increases as we use transceivers with longer tuning time (i.e., 360ns, 480ns, 600ns, 1200ns, corresponding to 30%, 40% 50%, 100% of the maximum transmission time of one packet). We conclude that in order to obtain a good performance of packet delay (e.g., less than 100 μs under the load of 0.6) with packet-level switching granularity in POTORI, T_{Tu} should be less than 30% of maximum transmission time of one packet.

With a longer transceiver's tuning time, T_M should be increased to reduce the tuning overhead. In Fig. 9 we present the packet delay as a function of τ, which is defined as the ratio of T_M over T_{Tu}, given T_{Tu} = 2 μs and 5 μs, respectively. With a small τ (i.e., τ = 1 and 2), the packet delay is as high as $10^4\mu$s even under the medium load (i.e., load = 0.4 and 0.6) while obviously better packet delay performance can be achieved with a larger τ (i.e., τ = 3, 4, 5) under the same load. In addition, with τ = 5, the performance will decrease a little at load = 0.6. This is due to the fact that even if with larger τ, more traffic can be sent in one cycle, yet not all servers cannot fully utilize the cycle and may transmit much less traffic than is allowed in one cycle. It is caused by the bursty feature of the traffic generated by the servers, which may lead to quite different traffic demand per server in each cycle. The larger the T_M is, the larger can be the difference in traffic requested by the servers in one cycle. In order to use the resources more efficiently and achieve the acceptable delay performance, we conclude that given a tunable transceiver with long tuning time (i.e., in the scale of micro-seconds), the maximum transmission time of one cycle should be at least three times longer than the transceiver's tuning time.

VII. Conclusion

In this paper, we focus on POTORI, an efficient optical ToR interconnect architecture for DCs. The POTORI's data plane is based on a passive optical coupler, which interconnects the tunable transceivers of the servers within a rack. In the control plane, POTORI relies on a centralized rack controller, which is responsible for managing the intra-rack and inter-rack communications. A cycle based centralized MAC protocol and a DBA algorithm (*Largest First*) are proposed and tailored for POTORI, aiming to achieve the collision-free transmission with good network performance. POTORI can be applied in optical DCN with any aggregation/core tier architecture, given the proper design of interfaces.

The simulation results have shown that under the realistic data center traffic scenarios, POTORI with the proposed DBA algorithm obtains the average packet delay in the order of microseconds, which is superior to the performance of the conventional EPS. Moreover, we quantify the impact of network configurations (including the interconnect size and the number of available wavelengths) and transceiver tuning time on the packet delay. For POTORI, the packet-level switching granularity is feasible if the tuning time can be kept small enough (less than 30% of the packet transmission time). In the case of the short tuning time (i.e., in the magnitude of microseconds), setting the maximum transmission time of each cycle greater than three times of transceiver's tuning time is still able to achieve the acceptable packet delay performance.

Acknowledgment

This work was supported by the Swedish Foundation for Strategic Research (SSF), Swedish Research Council (VR), Göran Gustafssons Stiftelse, Celtic-Plus subproject SENDATE-EXTEND funded by Vinnova, Natural Science Foundation of Guangdong Province (Grant No. 508206351021) and National Natural Science Foundation of China (Grant No. 61550110240, 61671212).

References

[1] http://www.datacenterknowledge.com/

[2] Cisco Global Cloud Index: Forecast and Methodology, 2014-2019, October 2015, Cisco white paper.

[3] A. Andreyev, "Introducing data center fabric, the next-generation Facebook data center network." https://code.facebook.com/posts/360346274145943, 2014.

[4] FINISAR, "Cabling in the Data Center", https://zh.finisar.com/markets/data-center/cabling-data-center.

[5] Data Center Design Considerations with 40 GbE and 100 GbE, Aug. 2013, Dell white paper.

[6] N. Binkert *et al*., "The role of optics in future high radix switch design," in *Proc. IEEE ISCA*, 2011, pp.437-447.

[7] R. Pries *et al*.,"Power consumption analysis of data center architectures," in *Green Communications and Networking*, 2012.

[8] M. Fiorani *et al*., "Energy-efficient elastic optical interconnect architecture for data centers," in *IEEE Communications Letters*, vol.18, pp. 1531-1534, Sept. 2014.

[9] R. Lin *et al*., "Experimental Validation of Scalability Improvement for Passive Optical Interconnect by Implementing Digital Equalization", European conference and exhibition on optical communication(ECOC), September 2016.

[10] G. Wan *et al*., "c-through: part-time optics in data centers," in *Proc. ACM SIGCOMM Conf.*, 2010, pp. 327-338.

[11] M. Fiorani *et al*., "Hybrid Optical Switching for Data Center Networks," in *Hindawi Journal of Electrical and Computer Engineering*, Vol. 2014, Article ID 139213, 13 pages, 2014.

[12] W. M. Mellette *et al.*, "A Scalable, Partially Configurable Optical Switch for Data Center Networks," in *IEEE/OSA Journal of Lightwave Technology*, vol. 35, no. 2, pp. 136-144, Jan. 2017.

[13] F. Yan *et al.*, "Novel Flat Data Center Network Architecture Based on Optical Switches With Fast Flow Control," in *IEEE Photonics Journal*, vol. 8, number 2, April 2016.

[14] M. Yuang *et al.*, "OPMDC: Architecture Design and Implementation of a New Optical Pyramid Data Center Network," in *IEEE/OSA Journal of Lightwave Technology*, vol. 33, issue 10, pages 2019-2031, May 2015.

[15] M. Fiorani *et al.*, "Optical spatial division multiplexing for ultra-high-capacity modular data centers," in *Proc. IEEE/OSA Opt. Fiber Commun. Conf.* 2016, Paper Tu2h.2

[16] V. Kamchevska *et al.*, "Experimental Demonstration of Multidimensional Switching Nodes for All-Optical Data Center Networks," in *IEEE/OSA Journal of Lightwave Technology*, vol. 34, issue 8, April 2016.

[17] A. Roy *et al.*, "Inside the Social Network's (Datacenter) Network," in *Proc. ACM SIGCOMM Conf.*, 2015 pp. 123-237.

[18] Y. Cheng *et al.*, "Centralized Control Plane for Passive Optical Top-of-Rack Interconnects in Data Centers," in *Proc. IEEE GLOBECOM* 2016.

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

[20] H. Yang *et al.*, "SUDOI: software defined networking for ubiquitous data center optical interconnection," IEEE Communications Magazine, vol. 54, no. 2, pp. 86-95, Feb. 2016.

[21] J. Chen *et al.*, "Optical Interconnects at Top of the Rack for Energy-Efficient Datacenters," in *IEEE Communications Magazine*, vol. 53, pp. 140-148, Aug. 2015.

[22] Y. Cheng *et al.*, "Reliable and Cost Efficient Passive Optical Interconnects for Data Centers," in *IEEE Communications Letters*, vol. 19, pp. 1913-1916, Nov. 2015.

[23] N. McKeown *et al.*, "OpenFlow: Enabling innovation in campus networks," in *ACM SIGCOMM Computer Communication*, Review 38, April 2008.

[24] Software-Defined Networking: The New Norm for Networks, Open Networking Foundation (ONF) White Paper, April 2012.

[25] "802.3-2012 - IEEE Standard for Ethernet".

[26] W. Ni *et al.*, "POXN: a new passive optical cross-connection network for low cost power efficient datacenters," in *IEEE/OSA Journal of Lightwave Technology*, vol. 32, pp. 1482-1500, Apr. 2014.

[27] "IEEE Standard for Information technology–Telecommunications and information exchange between systems Local and metropolitan area networks–Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications".

[28] L. Khermosh, "Managed Objects of Ethernet Passive Optical Networks (EPON)," RFC 4837, July 2007.

[29] B. Vattikonda *et al.*, "Practical TDMA for Datacenter Ethernet," in *Proc. ACM EuroSys. Conf.*, pp. 225-238, 2012.

[30] G. Poter *et al.*, "Integrating microsecond circuit switching into the data center," in *Proc. ACM SIGCOMM Conf.*, 2013 pp. 447-458.

[31] T. Javadi *et al.*, "A high-Throughput Algorithm for Buffered Crossbar Switch Fabric," in *Proceedings IEEE ICC*, pp. 1581- 1591, June 2001.

[32] S. Kandula *et al.*, "The Nature of Datacenter Traffic: Measurement &Analysis," in *Proc. ACM SIGCOMM Internet Eas. Conf.*, 2009, pp. 202-208.

[33] S. Matsuo *et al.*, "Microring-resonator-based widely tunable lasers," in *IEEE J. Sel. Topics Quantum Electron*, vol. 15, no. 3, pp. 545-554, 2009.

[34] Keysight Technology: 81960A Fast Swept Compact Tunable Laser Source, 1505nm to 1630nm
http://www.keysight.com/en/pd-2038953-pn-81960A/fast-swept-compact-tunable-laser-source-1505nm-to-1630nm-new?cc=SE&lc=eng

Yuxin Cheng received the B.Eng. degree in electronic engineering from Tsinghua University, Beijing, China, in 2012 and M. Eng. degree in network services and systems from KTH Royal Institute of Technology, Sweden, in 2015. He is currently a Ph.D. candidate with Optical Networks Lab (ONLab), KTH Royal Institute of Technology, Sweden. His research interests include optical data center network and software defined networking.

Matteo Fiorani received the Ph.D. degree (February 2014) in ICT from University of Modena (Italy). From February 2014 to October 2016, he was Postdoc Researcher in optical networks at KTH (Sweden). He was visiting researcher at TU Vienna (Austria), UC Davis (USA) and Columbia University (USA). Since October 2016, he works as Senior Researcher in 5G networks at Ericsson (Sweden). He co-authored more than 50 papers published in leading technical journals and conference proceedings. He also submitted several patents and standardization contributions on 5G networks. He was co-founder and chair of an IEEE workshop on 5G transport networks.

Rui Lin received the B.Sc. degree in Electrical Information from Huazhong University of Science and Technology (HUST), and the Ph.D. degree in Communication System from KTH Royal Institute of Technology, Sweden in 2016. Her research interests include short-reach optical interconnect and datacenter network.

Lena Wosinska received her Ph.D. degree in Photonics and Docent degree in Optical Networking from KTH Royal Institute of Technology, Sweden, where she is currently a Full Professor of Telecommunication in the School of Information and Communication Technology (ICT). She is founder and leader of the Optical Networks Lab (ONLab). She has been working in several EU projects and coordinating a number of national and international research projects.

Her research interests include fiber access and 5G transport networks, energy efficient optical networks, photonics in switching, optical network control, reliability and survivability, and optical datacenter networks. She has been involved in many professional activities including guest editorship of IEEE, OSA, Elsevier and Springer journals, serving as General Chair and Co-Chair of several IEEE, OSA and SPIE conferences and workshops, serving in TPC of many conferences, as well as being reviewer for scientific journals and project proposals. She has been an Associate Editor of OSA Journal of Optical Networking and IEEE/OSA Journal of Optical Communications and Networking. Currently she is serving on the Editorial Board of Springer Photonic Networks Communication Journal and of Wiley Transactions on Emerging Telecommunications Technologies.

Jiajia Chen received the Ph.D. degree (May 2009) in Microelectronics and Applied Physics, and the docent degree (February 2015) in Optical Networking from KTH Royal Institute of Technology, Sweden. Her main research interests are optical interconnect and transport networks supporting future mobile system and cloud environment. She has more than 100 papers published in the leading international journals and conferences. She is Principle Investigator of several national research projects funded by Swedish Foundation of Strategic Research (SSF), Gran Gustafssons Foundation and Swedish Research Council (VR). Meanwhile, she has been involved and taken the leadership in various European research projects, including the European FP7 projects IP-OASE (Integrated Project-Optical Access Seamless Evolution) and IP-DISCUS (Integrated Project-the DIStributed Core for unlimited bandwidth supply for all Users and Services), and several EIT-ICT projects (e.g., Mobile backhaul, M2M and Xhaul).