# A Cross-Layer SDN Control Plane for Optical Multicast-Featured Datacenters

Yiting Xia and T. S. Eugene Ng
Dept. of Computer Science, Rice University
Houston, TX, USA

## Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design

## Keywords

Software Defined Networking; Control Plane; Multicast; Optical Networking; Data Center Architecture

## 1. INTRODUCTION

The increasing number of datacenter applications with heavy one-to-many communications has motivated the proposal of optical multicast-featured datacenter architectures. These solutions use optical power splitters to duplicate the optical signal from the input port to all the output ports on the fly, thus are promising for fast, reliable, economical, and energy-efficient group data delivery.

Figure 1 illustrates how optical power splitters can be inserted into a datacenter. In case of heavy multicast communications from a particular sender to a group of receivers, an optical power splitter can be connected to these servers statically. They can also be placed to the level of Top-of-Rack (ToR) switches to aggregate traffic, so that a set of servers beneath the sender ToR switch can multicast to another set of servers across the destination ToR switches. Dynamic allocation of optical power splitters can be achieved by having a high-radix optical space switch, e.g. a 3D MEMS switch, as a connectivity substrate [5].

The key challenge of these architectures lies in fitting optical data duplication in the picture of the existing network stack. Optical power splitters can only send data unidirectionally, from input to output but not vice versa. Today's unicast routing protocols assume bidirectional links and thus are ignorant of these additional optical paths. Since multicast routing protocols depend on unicast routing protocols for topology discovery, multicast trees will not be built over the optical splitters.

Utilizing the optical multicast function requires multicast data delivery to be performed over the optical power splitter while all other traffic for maintaining the multicast tree to be transmitted through paths elsewhere. This asymmetric routing behavior can be achieved by either a specialized multicast routing protocol or SDN control mechanisms.

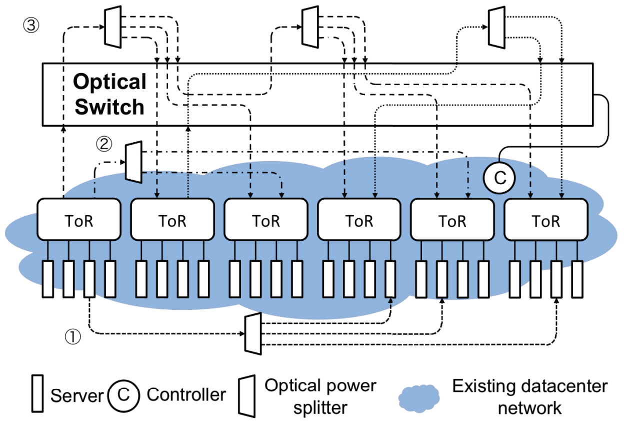

**Figure 1: Optical power splitters in a datacenter: ① static connections to servers; ② static connections to ToR switches; ③ dynamic connections to ToR switches via an optical switch**

The SDN approach has advantages across the network layers. In the physical layer, it has global view of the topology, while a specialized multicast routing protocol needs complicated discovery mechanisms to learn the locations of the optical power splitters. In the network layer, an SDN-controlled system can leverage prior knowledge of group membership to build a multicast tree instantaneously, whereas a multicast routing protocol takes efforts to maintain the group membership. In the application layer, an SDN controller is able to ensure the application performance through QoS negotiations, while QoS control in a distributed system is complicated.

In this paper, we build a cross-layer SDN control plane that leverages these advantages. We demonstrate the efficacy of the system by a Mininet prototype. Our control plane implementation is portable to a real datacenter.

## 2. CONTROL PLANE DESIGN

### 2.1 Network API

Prior studies of optical networked systems suggest application awareness is necessary for circuit utilization and ap-

*HotSDN'14,* August 22, 2014, Chicago, Illinois, USA.
ACM 978-1-4503-2989-7/14/08.
http://dx.doi.org/10.1145/2620728.2620778 .

plication performance [2][4]. In our multicast-featured system, we further leverage application awareness to obtain group memberships and build optical multicast trees rapidly.

Application managers, such as the Spark master node [6], can request for optical resources explicitly through our network API. Specifically, application managers inform the SDN controller of the multicast group memberships and traffic demands. If a request gets accepted, some optical resources are dedicated for an estimated service time. If the multicast transmission is not completed during this period, the application manager can extend the request with the residual traffic volume. Because optical resources are limited and not all requests can be serviced immediately, issued requests can be withdrawn at any time.

## 2.2 Control Algorithm

Dynamic allocation of optical power splitters can be formulated as a maximum weighted hypergraph $b$-matching problem [3]. The multicast requests form a hypergraph $H = (V, E)$. $V$ is the set of vertices, where each vertex represents a rack; and $E$ is the set of hyperedges connecting any number of vertices, where each hyperedge embodies all the racks involved in the multicast. The weight of a hyperedge is the multicast traffic demand. We seek a maximum weight sub-collection of hyperedges such that each vertex is met by at most $b$ hyperedges, where $b$ is the number of optical ports per rack.

Hypergraph matching is NP-hard [3]. We develop a greedy algorithm to solve the problem. The multicast requests are sorted by decreasing traffic demand. Then the algorithm iteratively selects the requests with the greatest demand as long as every involved rack has appeared in less than $b$ previously selected requests. The algorithm continues until no more requests can be added with all the constraints satisfied.

## 2.3 Reconfiguration

The ToR switches or end servers that the optical power splitters attach to must support SDN, e.g. OpenFlow switches or Open vSwitch instances. The network controller sets OpenFlow rules on them to direct multicast traffic through the constructed multicast trees. For dynamic allocation of optical splitters, the network controller also changes the optical switch interconnections through the command-line interface, such as TL1.

## 3. MININET IMPLEMENTATION

We implement the SDN control plane in Mininet, which is directly portable to a real datacenter network. The application-network communication procedures can be realized using cross-language RPC frameworks, because they provide well-defined APIs for various programming languages and a complete network stack with optimized end-to-end performance. With a potentially large number of requests, the controller is non-blocking to incoming connections and are able to handle concurrent requests. In case of frequent multicast requests, the control algorithm can serve in batch mode. Our controller sets OpenFlow rules for the multicast tree construction. It may need to reconfigure the optical switch to set up optical paths for the scheduled multicast groups. Because the switching latency of an optical circuit switch is not negligible (tens of ms), OpenFlow rules can be set simultaneously with the optical switch reconfiguration to reduce the overall delay.

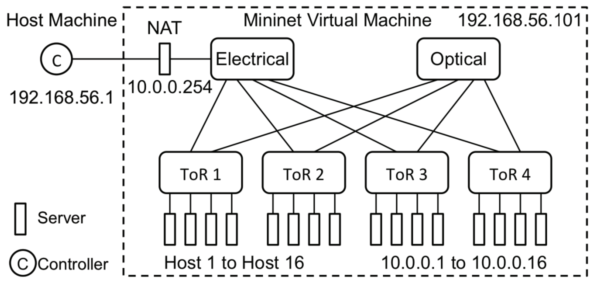

**Figure 2: Mininet virtual network structure**

Finally, the application is notified through the RPC call back function and the multicast transmission can start.

Figure 2 plots the structure of our Mininet virtual network. Mininet runs as a virtual machine on a host machine. The datacenter network core is abstracted by a single switch, denoted as "Electrical"; the optical switch together with the attached optical power splitters are emulated by another switch, denoted as "Optical". Our control plane software runs on the host machine. A NAT is set up to enable the hosts with non-routable internal IP addresses to access the host machine. The control algorithm is configured to compute every 1 second. End hosts run a multicast application built from a reliable multicast library [1].

We observe that the requests are returned rapidly, less than 20 ms after the control algorithm starts computation. We further break down the latency of a request processing: 1.747 ms for application-controller communication, 8.553 ms for computation, 7.711 ms for setting OpenFlow rules. Though the Mininet virtual network is different than a real datacenter, these results partially demonstrate the responsiveness of the control plane software.

## 4. REFERENCES

[1] JGroups - A Toolkit for Reliable Multicast Communication, http://www.jgroups.org/.

[2] H. H. Bazzaz, M. Tewari, G. Wang, G. Porter, T. S. E. Ng, D. G. Andersen, M. Kaminsky, M. A. Kozuch, and A. Vahdat. Switching the Optical Divide: Fundamental Challenges for Hybrid Electrical Optical Datacenter Networks. In *SOCC '11*, pages 1–8, Cascais, Portugal, Oct. 2011.

[3] O. Parekh. Iterative Packing for Demand and Hypergraph Matching. In *IPCO'11*, pages 349–361, Berlin, Heidelberg, 2011. Springer-Verlag.

[4] G. Wang, T. S. E. Ng, and A. Shaikh. Programming Your Network at Run-time for Big Data Applications. In *HotSDN '12*, pages 103–108, Helsinki, Finland, Aug. 2012.

[5] H. Wang, Y. Xia, K. Bergman, T. S. E. Ng, S. Sahu, and K. Sripanidkulchai. Rethinking the Physical Layer of Data Center Networks of the Next Decade: Using Optics to Enable Efficient *-Cast Connectivity. *SIGCOMM Computer Communication Review*, 43(3):52–58, July 2013.

[6] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica. Spark: Cluster computing with working sets. In *Proceedings of the 2Nd USENIX Conference on Hot Topics in Cloud Computing*, HotCloud'10, pages 10–10, Berkeley, CA, USA, 2010. USENIX Association.