CHALMERS

Chalmers Publication Library

Multicast Scheduling for Optical Data Center Switches with Tunability Constraints

This document has been downloaded from Chalmers Publication Library (CPL). It is the author´s version of a work that was accepted for publication in:
International Conference on Computing, Networking and Communications (ICNC), Silicon Valley, USA, Jan 2017,

Citation for the published paper:
Keykhosravi, K. ; Rastegarfar, H. ; Agrell, E. (2017) "Multicast Scheduling for Optical Data Center Switches with Tunability Constraints". International Conference on Computing, Networking and Communications (ICNC), Silicon Valley, USA, Jan 2017, pp. 5.

Downloaded from: http://publications.lib.chalmers.se/publication/250958

Notice: Changes introduced as a result of publishing processes such as copy-editing and formatting may not be reflected in this document. For a definitive version of this work, please refer to the published source. Please note that access to the published version might require a subscription.

Chalmers Publication Library (CPL) offers the possibility of retrieving research publications produced at Chalmers University of Technology. It covers all types of publications: articles, dissertations, licentiate theses, masters theses, conference papers, reports etc. Since 2006 it is the official tool for Chalmers official publication statistics. To ensure that Chalmers research results are disseminated as widely as possible, an Open Access Policy has been adopted.
The CPL service is administrated and maintained by Chalmers Library.

(article starts on next page)

Multicast Scheduling for Optical Data Center Switches with Tunability Constraints

Kamran Keykhosravi, Houman Rastegarfar, and Erik Agrell
Department of Signals and Systems, Chalmers University of Technology, 412 96 Gothenburg, Sweden
Email: {kamrank,houman.rastegarfar,agrell} @chalmers.se

***Abstract*—Optical multicasting based on passive star couplers and fast tunable transceivers is an attractive solution for the throughput and latency requirements of many data center applications. The limited tuning range of transceivers, however, may not be sufficient enough to enable the flexible scheduling of traffic. In this paper, we propose a suite of scalable scheduling algorithms for optical multicast switches with wavelength tunability constraints, considering both tunable and nontunable transmitters. To support scalability and scheduling fairness, we adopt a round-robin arbitration policy in conjunction with appropriate provisions to minimize the number of packet retransmissions. We conduct Monte Carlo simulations to compare the proposed algorithms. For 64 ports, 16 channels, and bursty multicast traffic, a scheduling that exploits transmitter tunability with minimal fan-out splitting can improve the maximum throughput by up to 60% compared to a fixed transmitter scenario.**

I. Introduction

Many applications in modern data centers call for multicast traffic delivery, i.e., the transmission of the same copy of information to several recipients simultaneously. Application examples include directing search queries to a set of indexing servers and distributing executable binaries to a group of servers participating in cooperative computations such as MapReduce [1]. Multicast benefits such applications in two distinct aspects. First, by minimizing network load, it increases the throughput of bandwidth-hungry computations. Second, by preventing multiple transmissions of the same packets to different receivers, it helps to reduce the completion time of delay-sensitive applications [1]–[4]. Multicasting can further save on network communication resources and energy requirements which are significant concerns in cloud data centers [3].

In implementing efficient and scalable multicast schemes for data centers, one should take into account the switching hardware capabilities and requirements. In traditional electronic data center networks, IP multicast is the most prevalent solution for one-to-many transmission, requiring complex hardware configuration [1], [4], [5]. The lack of proper multicast protocols in existing data centers is accompanied with the overwhelming challenges of electronic switching in data centers in terms of scale, footprint, cable management, and power consumption. Optical switching supporting massive scales, bit rate transparency, and low energy footprints not only helps to alleviate the electronic switching bottlenecks in data centers, but can also be utilized for high-capacity and energy-efficient traffic multicasting [5]–[7]. Recently, it has been shown that replacing electronic top-of-rack (ToR) switches with multicast-enabled optical switches can lead to significant energy savings in data centers [7].

In this paper, we consider the problem of multicast scheduling of optical packets in a data center subject to laser tunability constraints. We consider the architecture presented in Fig. 1 throughout our study. The switch comprises an $N \times N$ star coupler interconnecting N servers (or computing nodes). The switch can be located at different tiers of the data center, including top of racks. Each node is equipped with a fast tunable transceiver (capable of tuning over $W \leq N$ wavelengths in 10's of nanosecond time-scale) for transmitting and receiving data packets and a small form-factor pluggable (SFP) transceiver for interfacing with the switch controller (scheduler). Packets are stored in a single queue within each node (i.e., edge buffering). Depending on traffic demands, the scheduler instructs nodes to tune to proper wavelengths for unicast or multicast transmission. As the coupler realizes a shared transmission medium, each transmitted signal is routed to all output ports. Hence, it is the responsibility of the intended recipients to properly tune their reception wavelengths.

The multicast switch scheduler should ensure high application throughput, low latency, and fair operation. It should arbitrate transmissions such that no collisions take place in the shared switch fabric. If the number of wavelengths, W, is equal to the switch port count, it will be possible for all transmitters to transmit concurrently (with each tuned to a distinct wavelength and targeting a unique receiver). However, in practice the number of wavelengths could be much smaller than N, due to potentially a large coupler size and/or fabrication costs and constraints. For instance, a 128 × 128 star coupler [8, Ch. 4] requires transceivers with 50.8 nm (i.e., 127×0.4 nm) tuning range should the 50 GHz ITU grid be considered. A limited number of channels degrades transmission flexibility and poses scheduling constraints. As a result, smart and yet simple-to-scale algorithms become crucial for supporting traffic demands as well as minimizing the impact of hardware constraints.

The problem of multicast scheduling for packet switches

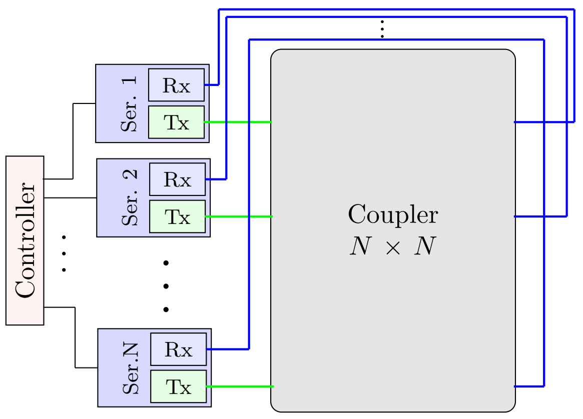

Fig. 1: Network interconnection based on an $N \times N$ coupler.

has received attention in a variety of contexts [9]–[16]. Although several scheduling algorithms have been proposed for different packet switches, how to efficiently schedule multicast packets is still a challenging issue. A group of proposals, for instance [9]–[12], deal with scheduling packets within electronic switches by resorting to different packet buffering strategies. They only consider nonblocking electronic switches that support simultaneous all-to-all communications. On the other hand, existing efforts on optical multicast scheduling are either based on restrictive assumptions or impose huge computational burdens on the controller. For instance, the algorithms in [13], [14] are only appropriate for fixed transmission wavelengths whereas the random schemes in [15] assume a fixed multicast group size (i.e., fan-out). The hybrid multicast scheduling algorithms in [16] assume transmitter tunability and varied fan-out. Nonetheless, they are reservation-based and require the controller to handle a large set of status information. Not only our proposed algorithms in this paper are based on practical assumptions, but also improve system scalability by adopting a round-robin decision-making logic.

The rest of this paper is organized as follows. Section II presents three multicast scheduling algorithms that we propose for different tunability constraints. We propose an algorithm for the case of fixed transmitters and two algorithms for transmitters with limited tunability. Section III details the performance analysis framework and discusses our simulation results, considering uniform and bursty traffic patterns. Finally, Section IV concludes the paper.

II. Scheduling Algorithms

We present three scalable multicast scheduling algorithms for the switch in Fig. 1. Fig. 2 serves as an example of traffic demands within a 4×4 switch, where the age and destination set of head of line (HOL) packets are specified. We use this example to illustrate the scheduling steps of the each algorithm. Our first algorithm assumes nontunable transmitters and serves as a benchmark for quantifying the performance gains of other algorithms.

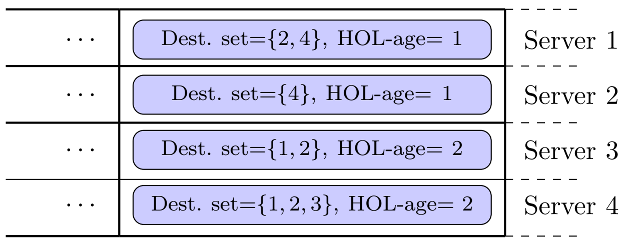
. . .
Dest. set=$\{2,4\}$, HOL-age= 1
Server 1
. . .
Dest. set=$\{4\}$, HOL-age= 1
Server 2
. . .
Dest. set=$\{1,2\}$, HOL-age= 2
Server 3
. . .
Dest. set=$\{1,2,3\}$, HOL-age= 2
Server 4

Fig. 2: An example of the HOL status for a 4×4 input-buffered multicast switch.

A. Algorithm 1: Weight-Based Algorithm for Nontunable Transmitters (WANT)

We assume that transmitters are assigned fixed wavelengths and cannot tune to other channels. Specifically, transmitter i , $1 \leq i \leq N$ is assigned wavelength $(i-1 \mod W)+1$. For simplicity, we assume that N is an integer multiple of the number of channels, W. Let g_k be the set of all nodes that are fix-tuned to the kth wavelength, i.e., $\mathsf{g}_k = \{\text{Node}(i) : (i-1 \mod W)+1 = k\}$, where $1 \leq k \leq W$. The proposed algorithm consists of two steps. First, selecting one server in every subset g_k, and second, scheduling the HOL packets of the selected servers to be transmitted in the next time slot. The former is preformed using a round-robin algorithm and the latter by a distributed weight-based algorithm as in [9]. The tasks preformed by the controller are listed as follows.

WANT scheduling algorithm:

1) *Determine the HOLs:* For every server, determine the destination(s) and the age of their HOL packet.
2) *Select servers:* For every set g_k, select the first server with nonempty queue. The initial point for the search changes in circular order and is determined by a round-robin pointer. For the example in Fig. 2, if $W = 4$, all of the four servers are selected. For $W = 2$, we have $\mathsf{g}_1 = \{1,3\}$ and $\mathsf{g}_2 = \{2,4\}$. In this case, if $Pointer = 1$ ($Pointer = 2$), the selected servers will be 1 and 2 (3 and 4). For the subsequent steps we assume $W = 4$ which makes every server present in the process.
3) *Assign weights to the HOL packets:* Based on HOL age and fan-out, assign a weight to each HOL packet. Specifically, calculate the weight as
$$w(s) = \text{HOL-age}(s) + f \times \text{Fan-out}(s) \tag{1}$$
where $1 \leq s \leq N$. For instance, assuming $f = -1$, we have $w(1) = -1$, $w(2) = 0$, $w(3) = 0$, and $w(4) = -1$.
4) *Request:* Consider the request of each selected server to the destination(s) of its HOL packet, including the weight calculated in (1).
5) *Grant:* For each receiver, grant the highest-weighted request among all requests destined to it. Ties are broken randomly. Denoting the granted server by receiver i as $G(i)$, in our example, we obtain $G(1) = 3$ (since $w(3) > w(4)$), $G(2) = 3$, $G(3) = 4$, and $G(4) = 2$.

6) *Tune the receivers:* In order to receive the granted packet, tune the ith receiver to the transmit wavelength assigned to server $G(i)$. In our example, receivers 1 to 4 are tuned to wavelengths 3, 3, 4, and 2, respectively.
7) *Update:* Update the pointer by calculating its modulus N/W and incrementing the result by 1. Also, update the HOL packet and HOL age.

After the completion of scheduling steps, the ith receiver receives the HOL packet of the transmitter $G(i)$.

B. Algorithm 2: Greedy Multicast Algorithm (GMA)

GMA is a greedy scheduling algorithm in which server selection and HOL packet scheduling are carried out in a single step. The algorithm is greedy in the sense that it schedules a specific HOL packet to as many destinations as possible. The process stops if all channels are used, all servers are checked, or all receivers are occupied. A pointer, which is updated in a round-robin fashion, is used to mark the starting point of the scheduler.

GMA Scheduling Algorithm:

1) *Determine the HOLs.*
2) *Select servers and destinations:* Starting from the server marked by the pointer, consider the HOL packet of the first server with a nonempty queue. Let B be the intersection of its destination set and the set of free receivers. If B is nonempty, assign a wavelength to the transmitter. Furthermore, let B determine the set of receivers that receive the packet in the next time slot. Tune them to the same wavelength as the one assigned to the selected transmitting server. Repeat this process until all channels are used, or all receivers are occupied, or all servers are examined. In our example, if the pointer value is equal to 3 and $W = 4$, receivers 1 and 2 receive from server 3, receiver 3 from server 4, and receiver 4 from server 1.
3) *Update:* Update the pointer by calculating its modulus N and incrementing the result by one. Also, update the HOL packet and HOL age.

C. Algorithm 3: Greedy Algorithm with Minimizing Fan-out Splitting (GAMFS)

To better utilize the multicast capability of the switch fabric, GAMFS first tries to schedule the packets without fan-out splitting (i.e, transmitting the same multicast packet over several time slots) and then addresses the remaining receivers using step 2 of the previous algorithm, GMA.

GAMFS Scheduling Algorithm :

1) *Determine the HOLs.*
2) *Schedule packets without fan-out splitting:* Consider the HOL packet of the server selected by the round-robin pointer and find the intersection of the destination set and the set of free receivers. If the intersection is equal to the destination set, assign a wavelength to the server and tune the destination(s) to the same wavelength. Check the subsequent servers. If all the receivers are occupied, or all the wavelengths are used, stop and go to step 4. Otherwise, go to step 3. In our example, if $Pointer = 3$ and $W = 4$, the algorithm first selects server 3 to send its HOL packet to receivers 1 and 2. Since receivers 1 and 2 have already been occupied, the algorithm does not choose server 4 for transmission. Server 1 is also not selected because receiver 2 is not free. Finally, the algorithm selects server 2. Since receiver 3 is free yet, the algorithm moves to step 3.
3) *Fill in the void by fan-out splitting:* Preform step 2 of GMA to schedule the remaining idle receivers.
4) *Update:* Same as in algorithm GMA.

III. Performance Analysis

In this section, we analyze via simulations the delay performance of the architecture in Fig. 1, considering the three proposed multicast scheduling algorithms. For the sake of brevity, other performance measures such as buffer occupancy and the average number of transmissions per packet are not studied here. In our analysis, the buffer depth is assumed to be worth 1000 packets, which is large enough as long as the queues are stable. N is set to 64 and three values for W are considered, namely, 16, 32, and 64. A total number of one million time slots are simulated, the second half of which contributes to the results presented here.

Traffic Model: We consider two traffic types: Bernoulli and Bursty. In the Bernoulli traffic model, packets are generated with probability ρ in each instance independently, where $0 \leq \rho \leq 1$ is the average number of packets per server generated in one time slot.

We also use geometric distribution to capture the bursty nature of data center traffic. Although a variety of distributions have been proposed for modeling burstiness in data center networks, we consider the geometric distribution due to its simplicity and popularity in studying multicast systems (see for instance [9], [11]). Within this model, each server is either in the ON or the OFF state. No packet arrives during the OFF period. However, during the ON period, one packet is added to the buffer at the beginning of each time slot. All packets generated during one ON period have the same destination set. The duration of the ON period is determined by realizations of a geometric random variable, g_{on}, whose distribution is $\Pr\{g_{\text{on}} = n\} = p_{\text{on}}(1-p_{\text{on}})^{n-1}$, where $n = 1, 2, 3, \ldots$ and $0 < p_{\text{on}} \leq 1$. The expectation of this random variable is $E_{\text{on}} = 1/p_{\text{on}}$. Moreover, an average arrival rate ρ implies $\rho = E_{\text{on}}/(E_{\text{on}} + E_{\text{off}})$, where E_{off} is the mean value of the OFF period which is also assumed to be geometrically distributed. We assume $E_{\text{on}} = 16$.

In both traffic schemes, the fan-out distribution is assumed to be truncated geometric with probability mass function

$$\Pr\{\text{Fan-out} = n\} = \frac{(1-q)q^{n-1}}{1-q^{N-1}} \quad 1 \leq n \leq N-1 \quad (2)$$

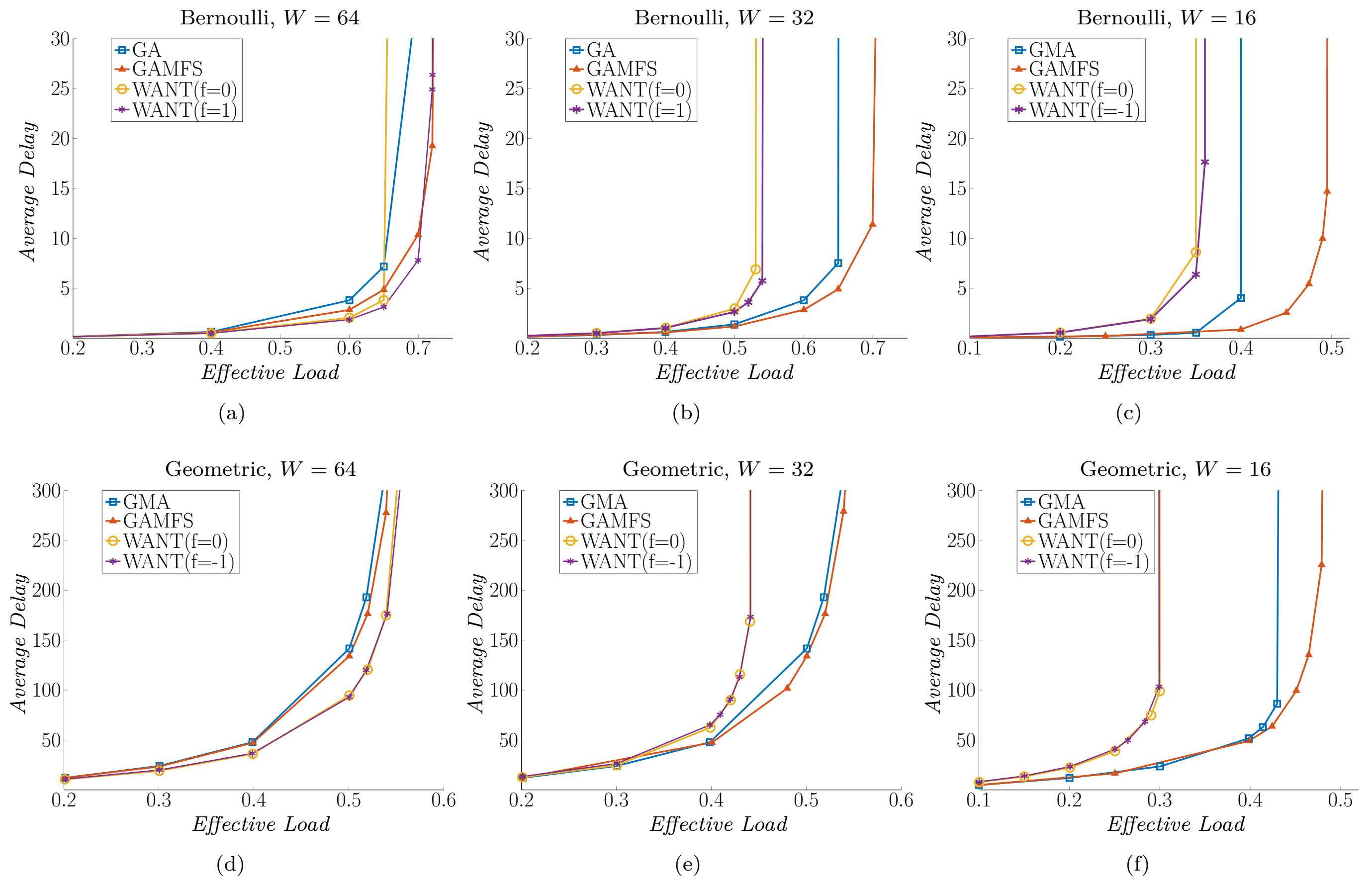

Fig. 3: Simulation results for a 64×64 switch. Average packet delay (in time slots) versus effective load for: (a) Bernoulli traffic and $W = 64$, (b) Bernoulli traffic and $W = 32$, (c) Bernoulli traffic and $W = 16$, (d) geometric traffic and $W = 64$, (e) geometric traffic and $W = 32$, and (f) geometric traffic and $W = 16$.

where $0 < q < 1$. The expected value of the fan-out is obtained by [11]

$$E[\text{Fan-out}] = \frac{1}{1-q} - \frac{(N-1)q^{N-1}}{1-q^N}. \quad (3)$$

We set $q = 0.5$ and note that servers are not allowed to generate packets destined to themselves.

Figure 3 depicts the simulation results for the two types of traffic (Bernoulli and geometric) and the three values of W (64, 32, and 16). In each setup, effective load (i.e., the average utilization of output ports) is depicted versus the average delay, (i.e., the average number of time slots a transmitted packet has waited in the queue). The results are plotted for the three proposed scheduling algorithms with two weights applied to algorithm WANT, namely, $f = 0$ and $f = -1$.

A work-conserving scheduling algorithm leaves an output port idle only if it is impossible to transmit a packet to it without interfering with existing scheduled transmissions. When $W < N$, among the three algorithms, GMA and GAMFS are work-conserving since their search ends only if they run out of channels, free receivers, or packets. However, WANT does not have this property since the selection of the transmitting servers is performed independently of their destinations. Therefore, a selected server may lose the contention leaving a channel unused while an unselected server could have used it to transmit its packet to an idle receiver. This is the main reason for the large performance gap between the weighted and greedy algorithms when $W < N$.

With Bernoulli traffic, one can observe that when $f = -1$, WANT performs better compared to $f = 0$. The authors in [9] show that for negative fan-out weights, a weight-based algorithm transmits a larger number of packets unabridged which counteracts the effects of HOL blocking, leading to a better performance. The smaller the value of f, the smaller the average delay. Since GAMFS aims to send as many intact packets as possible, the HOL-blocking effect is also reduced with this algorithm.

For Benoulli traffic and $W = 64$ (Fig. 3a), we observe that GAMFS and WANT($f = -1$) outperform GMA and WANT($f = 0$), which can be explained by the above discussion. For $W = 64$, WANT performs similarly to the weight-based algorithm proposed in [9], which is work-conserving. Not restricted by the number of channels, it outperforms GMA and GAMFS under a moderate traffic load.

When the number of channels is smaller than the switch port count ($W < N$), each selected server is desired to transmit to as many destinations as possible in order to maximize throughput. Therefore, reducing the amount of fan-out splitting increases the output port utilization. This is the main advantage of GAMFS compared to GMA.

For Bernoulli traffic and $W = 32$ (Fig. 3b), the greedy algorithms result in significant performance gains. Compared with $W = 64$, WANT experiences a large performance degradation when the number of resources is reduced by half, while the performance of GMA and GAMFS remain unchanged. The maximum throughput for WANT($f = -1$) is 0.54 and for GAMFS this value is 0.7, corresponding to a 30% gain.

Since in our simulation the average fan-out is approximately 2 (corresponding to a mix of unicast and multicast traffic), for $W = 16$ the effective load cannot exceed $2 \times 16/64 = 0.5$. In Fig. 3c we see that GAMFS almost achieves this limit, surpassing WANT which yields a maximum throughput of 0.36 by 39%. The advantages of minimizing fan-out splitting are more noticeable as the wavelength recourses become more scarce, leading to an increase in the gap between GAMFS and GMA.

For bursty traffic, since the load is distributed unevenly in time, the packets should wait longer in order to reach the HOL, resulting in significantly larger packet delays. Furthermore, with bursty traffic, back-to-back packets in queues often have the same fan-out. Therefore, there is not much opportunity for resolving HOL blocking. As can be seen in Figures 3d–3f, the performance for $f = 0$ and $f = -1$ are almost identical.

With bursty traffic and full wavelength tunability (i.e. $W = 64$) (Fig. 3d), the performance of WANT is slightly better than GAM and GAMFS. When the transceivers' tunability is restricted to 32 channels (Fig. 3e), unlike the greedy algorithms, the performance of WANT deteriorates noticeably. As GAMFS is a work-conserving algorithm and aims at minimizing fan-out splitting, for $W = 16$ (Fig. 3f), it almost reaches the 0.5 throughput upper bound and outperforms WANT by 60 percent.

Finally, we note that all three aforementioned algorithms are fair by bounding the maximum HOL age. Fairness is a significant issue in designing scheduling algorithms and ensures that no input gets starved for switching resources. Since in algorithms GMA and GAMFS, the transmission of the packet marked by the pointer is guaranteed, the HOL age is upper-bounded by $N - 1$. For WANT if $W = N$, the maximum HOL age is bounded by $N + |f|N' - 1$ [9], where N' is the maximum fan-out (in our case $N' = N - 1$). Furthermore, when $W < N$, since each server is selected at least once in every N/W time slots, the HOL-age can be upper-bounded by $N(N + |f|N' - 1)/W$.

IV. Conclusion

We proposed multicast scheduling solutions for optical interconnections with bandwidth-limited transceivers, including a weight-based distributed algorithm for nontunable transmitters and two greedy algorithms for tunable transmitters. Our simulations indicated that the performance of a design with fixed transmitters is significantly affected when the number of available wavelengths is decreased. A scheduling algorithm that avoids fan-out splitting offered the best performance. Our next step in this research involves the design of proper buffering schemes to minimize the traffic burstiness penalties. Future work should also investigate multicast scheduling for large-scale switches with multiple broadcast domains and limited wavelength tunability.

References

[1] D. Li, M. Xu, Y. Liu, X. Xie, Y. Cui, J. Wang, and G. Chen, "Reliable multicast in data center networks," *IEEE Transactions on Computers*, vol. 63, no. 8, pp. 2011–2024, Aug. 2014.

[2] Z. Guo, J. Duan, and Y. Yang, "On-line multicast scheduling with bounded congestion in fat-tree data center networks," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 1, pp. 102–115, Jan. 2014.

[3] W.-K. Jia, "A scalable multicast source routing architecture for data center networks," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 1, pp. 116–123, Jan. 2014.

[4] D. Li, Y. Li, J. Wu, S. Su, and J. Yu, "ESM: Efficient and scalable data center multicast routing," *IEEE/ACM Transactions on Networking (TON)*, vol. 20, no. 3, pp. 944–955, Jun. 2012.

[5] P. Samadi, V. Gupta, J. Xu, H. Wang, G. Zussman, and K. Bergman, "Optical multicast system for data center networks," *Optics Express*, vol. 23, no. 17, pp. 22 162–22 180, Aug. 2015.

[6] H. Wang, Y. Xia, K. Bergman, T. S. 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," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 3, pp. 52–58, Jul. 2013.

[7] J. Chen, Y. Gong, M. Fiorani, and S. Aleksic, "Optical interconnects at the top of the rack for energy-efficient data centers," *IEEE Communications Magazine*, vol. 53, no. 8, pp. 140–148, 2015.

[8] S. Shimada, *Coherent lightwave communications technology.* Springer Science & Business Media, 2012.

[9] B. Prabhakar, N. McKeown, and R. Ahuja, "Multicast scheduling for input-queued switches," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 5, pp. 855–866, 1997.

[10] S. Gupta and A. Aziz, "Multicast scheduling for switches with multiple input-queues," in *High Performance Interconnects, 2002. Proceedings. 10th Symposium on.* IEEE, pp. 28–33.

[11] D. Pan and Y. Yang, "FIFO-based multicast scheduling algorithm for virtual output queued packet switches," *IEEE Transactions on Computers*, vol. 54, no. 10, pp. 1283–1297, 2005.

[12] H. Yu, S. Ruepp, and M. S. Berger, "Multi-level round-robin multicast scheduling with look-ahead mechanism," in *2011 IEEE International Conference on Communications (ICC)*.

[13] G. N. Rouskas and M. H. Ammar, "Multidestination communication over tunable-receiver single-hop WDM networks," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 3, pp. 501–511, 1997.

[14] K. Naik, D. S. Wei, D. Krizanc, and S.-Y. Kuo, "A reservation-based multicast protocol for WDM optical star networks," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 9, pp. 1670–1680, 2004.

[15] E. Modiano, "Random algorithms for scheduling multicast traffic in WDM broadcast-and-select networks," *IEEE/ACM transactions on Networking*, vol. 7, no. 3, pp. 425–434, 1999.

[16] H.-C. Lin and C.-H. Wang, "A hybrid multicast scheduling algorithm for single-hop WDM networks," in *INFOCOM 2001. 20th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 1, pp. 169–178.