Silicon photonic-enabled bandwidth steering for resource-efficient high performance computing

Yiwen Shen, Madeleine Strom Glick, Keren Bergman

Yiwen Shen, Madeleine Strom Glick, Keren Bergman, "Silicon photonic-enabled bandwidth steering for resource-efficient high performance computing," Proc. SPIE 10946, Metro and Data Center Optical Networks and Short-Reach Links II, 1094605 (1 February 2019); doi: 10.1117/12.2509060

Event: SPIE OPTO, 2019, San Francisco, California, United States

Silicon Photonic-enabled Bandwidth Steering for Resource-Efficient High Performance Computing

Yiwen Shen, Madeleine Strom Glick and Keren Bergman

Lightwave Research Laboratories, Electrical Engineering Department, Columbia University, New York City, USA

ABSTRACT

This paper presents the integration of multiple silicon photonic (SiP) switches within a high-performance computing environment to enable network reconfigurability in order to achieve optimized bandwidth utilization. We demonstrate a physical testbed implementation that incorporates two fabricated SiP switches capable of switching traffic under real HPC benchmark workloads. The system uses dynamic optical bandwidth steering to match its physical network topology to the traffic characteristics of the application, and achieves up to approximately 40% reduction in application execution time of the high-performance computing benchmark. We present the detailed design of the network architecture and control plane of the system, and discuss the system performance improvements that arises from bandwidth-steering with silicon photonic-based circuit switching.

Keywords: High-performance computing, silicon photonics, optical switching

1. INTRODUCTION

The role of the interconnection network plays an increasingly important role in the performance of large-scale high-performance computing (HPC) systems. The stalling of raw computation power of individual processor cores and the demand for greater computation power have necessitated the parallelization of multiple CPU cores and other discrete components that are combined within vast network architectures.[1] The reliance on parallelism to produce performance growth requires the underlying network fabric to provide sufficient bandwidth resources to memory and storage peripherals required by the processor nodes to reach their full computational potential.

However, today's HPC systems follow a best-for-all approach that is characterized by over-provisioned, static networks. Such an approach is expensive and inefficient, and lacks the scalability to meet the rising demand in computational power.[2] Many HPC and data analytics applications feature skewed traffic matrices that concentrate traffic within a small percentage of the total available links, and this traffic pattern varies slowly over time.[3,4] Each application has an optimal topology, and operating an application with a skewed traffic pattern on a general-purpose, static network results in both under-utilized links that wastes energy consumption, as well as congested links that cannot feed processors with enough data resulting in sub-optimal system performance.[5]

These problems present a unique opportunity to use flexible networks that have the capabilities to perform physical layer topology reconfiguration for cluster-to-cluster communication. By allowing available network links to be dynamically switched to connect different endpoints, the network can flexibly steer the bandwidths of these links to match an application's unique traffic demands, in a process called bandwidth steering. High density and energy efficient bandwidth steering can be performed through the means of silicon photonic (SiP) switches, which can be fabricated in highly integrated platforms using manufacturing processes compatible with CMOS processes. This allows them to have many advantages over other forms of optical switching, such as low power consumption, low fabrication costs at large scales, and small area footprint compared to micro-electro mechanical systems (MEMS) actuated micro-mirror array-based switches, as well as the ability to be closely integrated with electronic drivers.[6]

The integration of SiP switches into a conventional electronic HPC system can enable network flexibility in an energy efficient and scalable manner. While there has been a great deal of work on various types of SiP switches intended for HPC and data center networks, most tend to focus heavily on the device properties itself,

Further author information: (Send correspondence to Y. Shen) E-mail: ys2799@columbia.edu

Metro and Data Center Optical Networks and Short-Reach Links II, edited by Atul K. Srivastava, Madeleine Glick, Youichi Akasaka, Proc. of SPIE Vol. 10946, 1094605 · © 2019 SPIE
CCC code: 0277-786X/19/$18 · doi: 10.1117/12.2509060

pushing the boundaries of switching speed, port count, insertion loss and crosstalk, but do not show their usage in an application setting beyond demonstrating its switching capabilities.[7–10] Works that focus on the network architecture present simulations with theoretical high-radix devices,[11, 12] or uses other forms of optical switching devices such as MEMs based switches.[13–15] This work seeks to address the middle ground by demonstrating the integration of SiP switching devices into an HPC network architecture, evaluating its feasibility and control mechanisms.

Specifically, we present a reconfigurable HPC network architecture based on low-port count SiP switches, which is an extension of our previous work.[6, 16] It features the integration of multiple fabricated microring-resonator (MRR) based SiP switches within a Dragonfly[17, 18] topology for dynamic bandwidth steering under real HPC benchmark workloads. The SiP switches connect different Dragonfly groups of electronic routers together, and do not add additional bandwidth; rather, they allow for a reconfigurable network that can optimize its usage of existing bandwidth resources based on the application's needs. The entire system is managed with a top-level software-defined networking (SDN) control plane, which coordinates all the active components in the system. On this testbed, we operate a skeletonized version of the Gyrokinetic Toroidal Code (GTC)[19] HPC benchmark application, and show improvements in system performance by matching the physical network topology to the application's traffic characteristics through appropriate switching of the SiP switches.

In Section 2 the network architecture and control plane will be described in detail. Section 3 presents the HPC testbed hardware setup, and Section 4 presents the experimental results. In Section 5 we discuss the implications of this work and its feasibility, comparison with existing solutions and next steps. Lastly in Section 6 the conclusion is presented.

2. NETWORK ARCHITECTURE AND OPERATION MECHANISM

2.1 Topology Reconfiguration for Dragonfly

The incorporation of SiP switches enables aggregate traffic from top-of-rack (ToR) or higher tiered electronic packet switches (EPS) to be carried and steered, which can be applied to many different static hierarchical network topologies such as Fat-tree or Dragonfly. This allows for maximized leveraging of the wavelength-division multiplexing (WDM) capabilities of optical switches to move large quantities of aggregate bandwidth from groups of servers in a simple manner. In this work, the base topology used is the Dragonfly, which partitions S routers into G groups, with each router connected to C compute nodes. The network connections are organized into two tiers - *local, intra-group* links connect routers of the same group, and *global, inter-group* links connect each group to each other. The canonical Dragonfly topology has an all-to-all topology where all routers in a group are fully connected, and each group also has a direct connection to every other group.

However, the advantage of high connectivity of Dragonfly topologies is offset by its diluted per-link bandwidth. Therefore, applications that have concentrated traffic results in suboptimal usage of link bandwidth resources, with some links highly congested while most links are heavily under-utilized. The insertion of SiP switches within the topology allows for control of the network topology to match the traffic matrices of the application. Specifically, it enables traffic in the inter-group links to be dynamically steered between different Dragonfly groups in order to change the amount of available links, and therefore total bandwidth available to each Dragonfly group. The routing algorithm used is assumed to be k shortest path routing. More details on bandwidth-steering for Dragonfly topology networks at scale is presented in our previous work.[4]

2.2 Switch Integration and Control

Figure. 1 illustrates the method of integrating the SiP switch within a conventional electronic network environment. An SDN control application manages all active components in the system, which consists of the ToR EPSs and the SiP switches. It controls the EPSs through OpenFlow, a standard southbound SDN API, which allows for management of the flow tables of each EPS which connects groups of servers. The SiP switch control is enabled with an in-house agent called the FPGA Controller, which translates commands in the form of Ethernet packets from the SDN controller for desired SiP switch ports to be connected into pre-defined electronic bias signals that are applied onto the SiP switch inputs using an FPGA interface. During a physical topology change, the SDN application simultaneously modifies the flow rule entries on the relevant EPS and the SiP switch configuration,

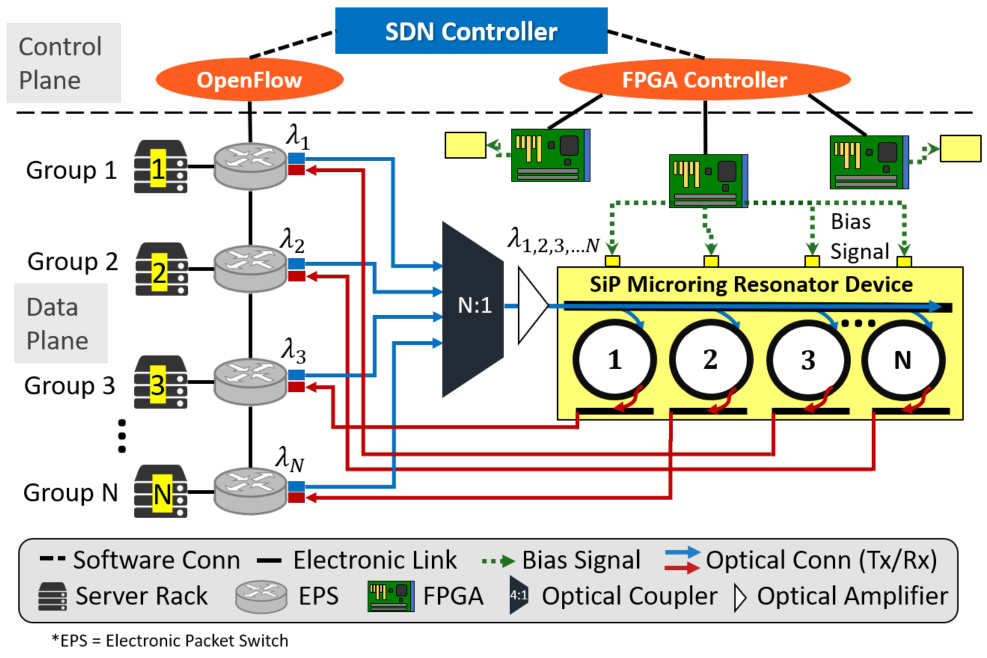

Figure 1. Network architecture showing the connections of the servers and top-of-rack EPSs to the silicon photonic MRR device.

allowing packets to reach from one server to another. The FPGA interface for SiP switch control consists of a distributed FPGA network that is a 1G out-of-band Ethernet network that can scale in a fat-tree topology to control larger SiP devices or multiple devices. Each FPGA is equipped with multiple digital-to-analog converter (DAC) chips, which is able to generate the appropriate voltage values that are stored in its registers. Electronic amplification of these bias signals are needed to provide enough power to cause the configuration change for the SiP device, ranging between 0 to 30 mW. Additional details of the control plane, including the communication protocols and FPGA operation procedure can be found in our previous work.[6]

To connect the SiP switch with the EPSs, each EPS transmits at a unique wavelength. Figure 1 shows N EPSs from N Dragonfly groups, while the SiP switch is a microring-resonator based SiP device with N rings. Each wavelength is multiplexed together and amplified before entering the optical waveguide of the SiP switch. Depending on how the microrings are tuned with the input bias signal, different wavelengths can be received on a chosen ring. Each ring's output connects back to the receiving end of the optical transceiver that is connected to the EPS.

3. SYSTEM TESTBED

3.1 Hardware Setup

The system testbed (Figure 2) is arranged in a Dragonfly topology consisting of 4 groups of 4 EPSs each, with 2 servers connected to each EPS. The EPSs are virtually partitioned from two physical Pica8 Ethernet switches with OpenFlow management capability, allowing for flow control and traffic monitoring using the SDN controller. The intra-group network is fully connected with 10G Direct-Attached copper cables, while the inter-group links are a combination of static electronic links and dynamic optical links with 10G DWDM SFP+ transceivers, with wavelengths C26, C28, C34, and C38. The optical links are dynamic as they are connected to the SiP switch(es), shown in the center. The fabricated SiP switches each feature four microring-resonators (MRRs), and these devices can be configured into one of three possible configurations with two inputs and two outputs, as shown in Figure 3. A single SiP switch replaces two static inter-group links, while the insertion of a second SiP switch adds two additional links and greatly improves the flexibility to which the network can adapt its topology to the traffic matrices. The resonance response of each MRR is separated by 1.27 nm with an FSR of 13 nm and each have a 3 dB bandwidth of 0.7 nm.

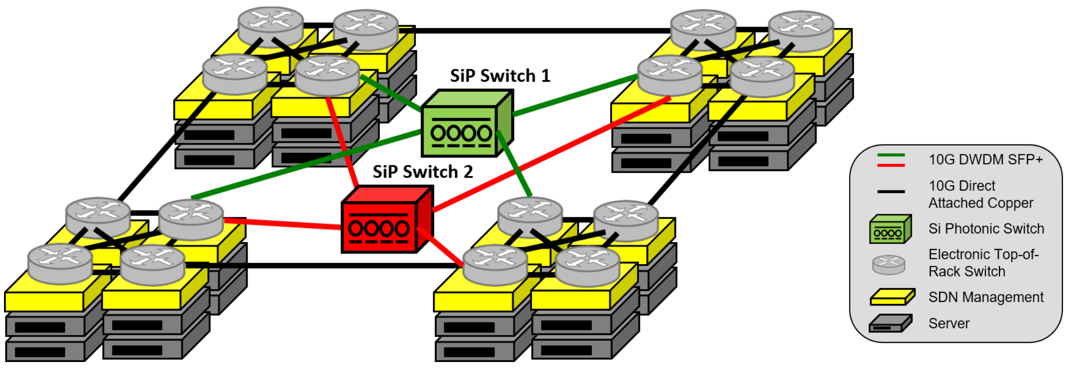

Figure 2. Testbed consisting of a Dragonfly topology and a silicon photonic MRR switch for inter-group reconfiguration.

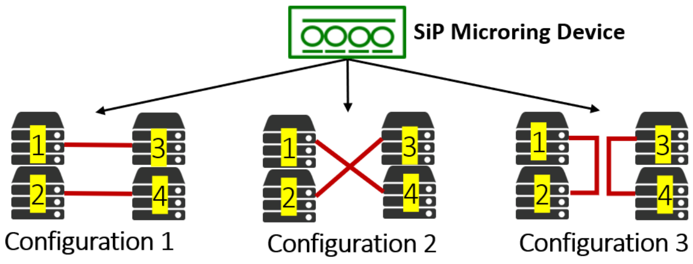

Figure 3. Each 4-ringed SiP microring device is capable of these three switching configurations

3.2 Software Benchmark Configuration

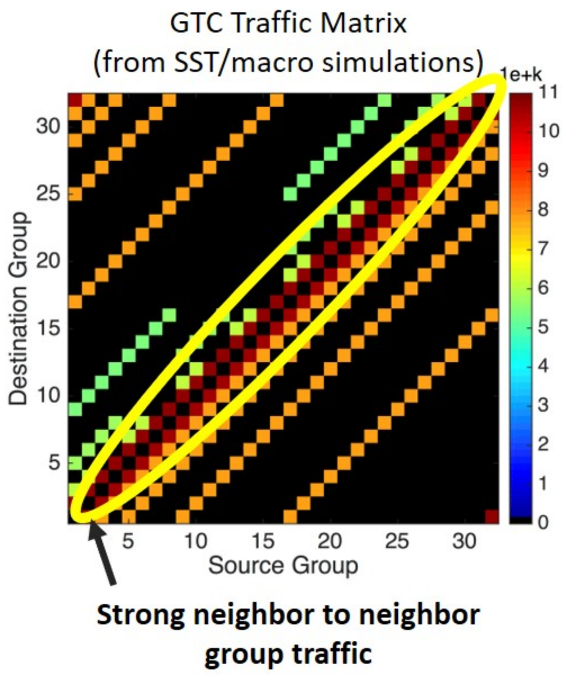

Figure 4. Traffic matrix of the GTC benchmark application

The testbed uses the MPI protocol for communication and synchronization of rank assignments over the physical machines. Specifically, MPICH[20] was used.

The application that is operated over the testbed is the Gyrokinetic Toroidal Code (GTC).[19] As can be seen in Figure 3.2 which shows a heat map of the traffic intensity between different source and destination Dragonfly groups obtained using simulations performed on the SST/macro, the yellow oval highlights the fact that the application displays strong +1/-1 neighbor to neighbor traffic, typical of many other HPC applications. We use a skeletonized version of the GTC benchmark code obtained from the public domain. The process of skeletonization is to remove computation routines while keeping the communication characteristics (packet sizes, destinations, and timing) the same. This allows for the application to run faster with the same communication pattern, which translates to a greater bandwidth demand over time. Greater bandwidth demand was necessary to create congestion in some of the links, in order to show the benefits in performance from relieving this congestion through bandwidth steering with the SiP switches.

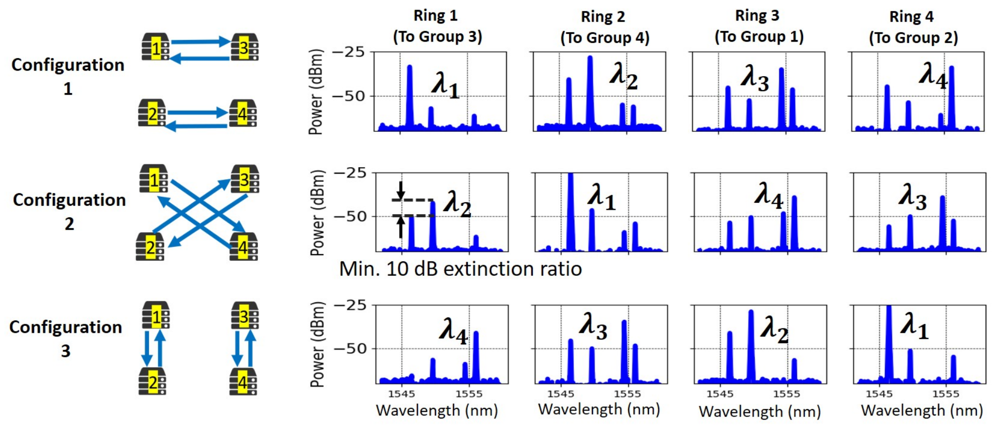

Figure 5. Spectra of the four input wavelength signals seen by the receiver side of the transceiver after being received by the microring

Table 1. Performance increase for various job sizes

Number of Ranks	Execution Time (s)		Performance Increase %
	Standard Dragonfly Topology	Bandwidth Steered Topology	
64	30	20	40%
128	46	30	42%
256	105	70	40%
512	252	186	30%

4. EXPERIMENTAL RESULTS

4.1 Optical Spectra

Figure 5 shows the spectra of the four input wavelengths for each switch configuration as seen by the receiver side of the commercial 10G SFP+ transceiver that is plugged into the EPS, without amplification. There are four peaks corresponding to the input signals from each group, and the highest peak is the desired signal that is received by tuning the microring resonator to that wavelength, while the other peaks are crosstalk. For the signal to be received properly, there is a minimum of 10 dB extinction ratio between the two highest peaks. The fiber-to-fiber loss of the SiP switch is approximately 10 to 15 dB, so that an optical amplifier is required to amplify the input signals before entering the SiP device.

4.2 Performance Improvement with a Single SiP Switch

In the first set of experiments, a single SiP switch is integrated into the testbed, and we ran various job sizes (Table 1) of our skeletonized GTC application through different number of ranks over a standard Dragonfly topology and a bandwidth-steered topology enabled by the SiP switch. For the skeletonized GTC application, increasing the number of ranks also increases the job size. To demonstrate the bandwidth steering process in more detail, the assignment of ranks to physical machines was done during runtime to purposely cause congestion between Groups 1 and 2, and between Groups 3 and 4. Figure 6 shows the throughput over time of the various inter-group links of the testbed network over the total execution time of the GTC application with 256 ranks. The top plot shows this for the standard Dragonfly topology. It can be seen that the links between Groups 1 and 2, and between Groups 3 and 4 are congested (red and blue), while the links between Groups 1 and Groups 4, as well as between Groups 2 and 3 show approximately zero usage, which matches the rank assignment. The execution time of GTC under this topology is 105 seconds.

Through the identification of the congested as well as the underutilized links from running the application over the standard all-to-all Dragonfly topology, it is clear that the links between Groups 1 to 4 and Groups 2 to 3 should be switched to relieve the congestion between Groups 1 and 2 and Groups 3 and 4, which is done in the bandwidth-steered topology. The result of running the same exact application is plotted on the lower

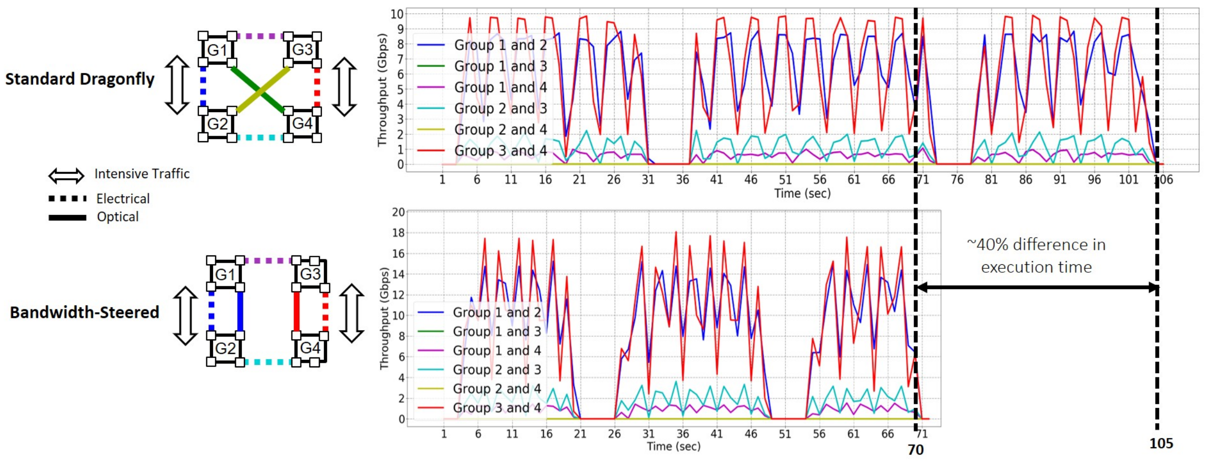

Figure 6. Throughput over time comparing a standard Dragonfly topology to a bandwidth-steered topology with a single SiP switch

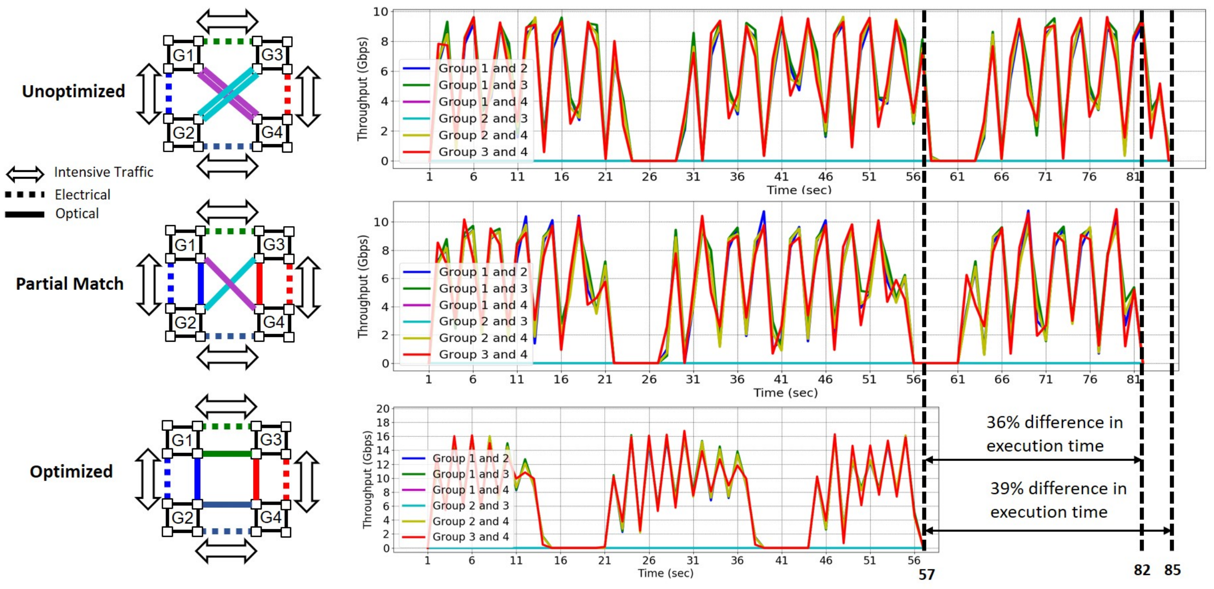

Figure 7. Throughput over time comparing different configured topologies enabled through two SiP switches

graph. Now with two links each between these pairs of communicating groups, the total bandwidth available between them is increased to a maximum of 20 Gbps, which is reflected in the y-axis of the plot. As can be seen, the traffic between Groups 1 and 2 and Groups 3 and 4 does indeed take advantage of this newly available bandwidth, and its throughput rises to 18 Gbps. We also align the time (x)-axis of the top and bottom plots and show that the increase in throughput due to bandwidth steering also allowed for the execution time of the application to be reduced by approximately 40%, from 105 seconds to 70 seconds.

4.3 Performance Improvement with Two SiP Switches

In the next set of experiments, two 4-ring SiP switches were inserted into the testbed network, in order to demonstrate the increased flexibility of the network to match its topology to the application traffic pattern. For this experiment, rank assignment to physical machines was done in a way to create intensive traffic between immediate neighboring groups, matching the real GTC traffic matrix shown in Figure 3.2. Under this traffic, we set up three different topologies as shown in Figure 7 - *unoptimized, partial match, and optimized* topologies, and run the GTC application to see the difference in system performance that is achieved due to bandwidth steering.

In the *unoptimized* topology, the two switches are both set to cross configuration, which does not help support the traffic pattern that is occurring on the outer edges. It leads to the longest execution time of 85 seconds. In the *partial match* topology, one of the SiP switches is configured to match the traffic, while the other one remains in the cross configuration. This allows for 20 Gbps of bandwidth to be available between Groups 1 and 2, and Groups 3 and 4. Yet despite this newly available bandwidth, the traffic barely takes advantage of it. It can be seen that some spikes in the traffic throughput do rise above 10 Gbps, but it does not use nearly the full capacity available. We theorize that this is due to the 10 Gbps bottleneck that is present between Groups 1 and 3, and Groups 2 and 4. Since computations must wait for nodes communicating between these groups, it slows down traffic in other parts of the system despite higher available network bandwidth. This is reflected in the application execution time of 82 seconds, which is only approximately 3% lower than for the *unoptimized* topology. Lastly in the *optimized* topology, both switches have been configured to match the traffic pattern, allowing for 20 Gbps bandwidth on all sides. This time there is a clear benefit, and the traffic rises to more than 16 Gbps, and the execution time is reduced to 57 seconds, a 39% reduction compared to the *unoptimized* topology and a 36% reduction compared to the *partial match* topology.

5. DISCUSSION

Through the experimental demonstrations presented in the previous section, it has been shown that the bandwidth steering concept is feasible for a conventional electronic system and can be realized using low-radix SiP switches. The addition of a second SiP switch to the system also shows that greater flexibility of the network is needed through the *partial match* network topology, which can be primarily achieved using a single switch. However as the results have shown, it would not have been enough to create a noticeable difference in system performance improvement. In addition it shows that provisioning additional bandwidth between nodes does not necessarily mean that the traffic throughput will increase, if there are bottlenecks elsewhere in the network. Lastly, through the demonstrations of these experiments, we can conclude that not only can the execution time of the system be significantly affected by the bandwidth available between each processing node, but that it is also proportional to the amount of throughput increase that is gained through bandwidth steering.

One may question the of use of SiP switches over an electronic packet switch for bandwidth steering. The primary motivation for using SiP switches over an EPS is the fact that the SiP switch can not only take advantage of the WDM capability of optical communications to carry a large amount of channels from electronic router groups, but also switch them in a simple manner that requires little changes on the software side. The network architecture features a flat topology where the SiP switches are layer 1 devices that are invisible to the electronic system, and the only part that is affected is the flow rules on the EPSs. In terms of using SiP switches over other types of optical switches such as MEMS, SiP switches are limited in their port count but have much lower energy consumption, cost, and footprint. However their main advantage over MEMS switches as stated in the Introduction is their ability to be closely integrated with electronic control drivers and other peripherals. Although in this demonstration the control interface uses FPGAs, the ideal system would be specialized application-specific integrated circuits (ASICs) that can be interfaced with a central control system, such as an SDN controller. The low port count of SiP switches for bandwidth steering in the Dragonfly topology is discussed in our previous work.[4]

In this work the rank assignment was chosen specifically to demonstrate the benefits that can be achieved with bandwidth steering. An accurate assessment of the flexible topologies enabled with one or two SiP switches would require evaluation of application execution time reduction over many randomized rank assignments, which will be part of our next steps. Other avenues to pursue are to investigate algorithms for bandwidth steering for a variable number of Dragonfly groups, routers per group, and different number of switch ports, as well as application of the bandwidth steering concept to other topologies such as the Fat-tree.

6. CONCLUSION

In this paper we have shown a reconfigurable HPC network architecture for the Dragonfly topology, through the integration of low-radix SiP switches for dynamic bandwidth steering. We demonstrate our concept with a physical testbed of 16 physical nodes and an SDN control plane managing two microring-resonator based SiP switches, and showed up to 40% system performance improvements through application execution time reduction, over topologies with one and two switches when running real HPC workloads with MPI.

ACKNOWLEDGMENTS

AIM Datacom; U.S. Department of Energy (DoE) Advanced Simulation and Computing (ASC) program through contract PO 1319001 with Sandia National Laboratories; Advanced Research Projects Agency Energy (ARPA-E) under the Enlightened Project.

REFERENCES

[1] Kogge, P. M. and Shalf, J., "Exascale computing trends: Adjusting to the "new normal"' for computer architecture," *Computing in Science and Engineering* **15**, 16–26 (2013).

[2] Rumley, S., Bahadori, M., Polster, R., Hammond, S. D., Calhoun, D. M., Wen, K., Rodrigues, A., and Bergman, K., "Optical interconnects for extreme scale computing systems," *Parallel Comput.* **64**, 65–80 (May 2017).

[3] Kamil, S., Pinar, A., Gunter, D., Lijewski, M., Oliker, L., and Shalf, J., "Reconfigurable hybrid interconnection for static and dynamic scientific applications," in [*Proceedings of the 4th International Conference on Computing Frontiers*], *CF '07*, 183–194, ACM, New York, NY, USA (2007).

[4] Wen, K., Samadi, P., Rumley, S., Chen, C. P., Shen, Y., Bahadori, M., Bergman, K., and Wilke, J., "Flexfly: Enabling a reconfigurable dragonfly through silicon photonics," in [*SC '16: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*], 166–177 (Nov 2016).

[5] Bhatele, A., Jain, N., Livnat, Y., Pascucci, V., and Bremer, P., "Analyzing network health and congestion in dragonfly-based supercomputers," in [*2016 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*], 93–102 (May 2016).

[6] Shen, Y., Hattink, M. H. N., Samadi, P., Cheng, Q., Hu, Z., Gazman, A., and Bergman, K., "Software-defined networking control plane for seamless integration of multiple silicon photonic switches in datacom networks," *Opt. Express* **26**, 10914–10929 (Apr 2018).

[7] Aguinaldo, R., Forencich, A., DeRose, C., Lentine, A., Trotter, D. C., Fainman, Y., Porter, G., Papen, G., and Mookherjea, S., "Wideband silicon-photonic thermo-optic switch in a wavelength-division multiplexed ring network," *Opt. Express* **22**, 8205–8218 (Apr 2014).

[8] Yu, R., Cheung, S., Li, Y., Okamoto, K., Proietti, R., Yin, Y., and Yoo, S. J. B., "A scalable silicon photonic chip-scale optical switch for high performance computing systems," *Opt. Express* **21**, 32655–32667 (Dec 2013).

[9] Gazman, A., Browning, C., Bahadori, M., Zhu, Z., Samadi, P., Rumley, S., Vujicic, V., Barry, L. P., and Bergman, K., "Software-defined control-plane for wavelength selective unicast and multicast of optical data in a silicon photonic platform," *Opt. Express* **25**, 232–242 (Jan 2017).

[10] Chu, T., Qiao, L., Tang, W., Guo, D., and Wu, W., "Fast, high-radix silicon photonic switches," in [*Optical Fiber Communication Conference*], *Optical Fiber Communication Conference* , Th1J.4, Optical Society of America (2018).

[11] Nikolova, D., Rumley, S., Calhoun, D., Li, Q., Hendry, R., Samadi, P., and Bergman, K., "Scaling silicon photonic switch fabrics for data center interconnection networks," *Opt. Express* **23**, 1159–1175 (Jan 2015).

[12] Kiyo Ishii, Takashi Inoue, S. N., "Toward exa-scale optical circuit switch interconnect networks for future datacenter/hpc," (2017).

[13] Xia, Y., Sun, X., Dzinamarira, S., Wu, D., Huang, X. S., and Ng, T. S. E., "A tale of two topologies: Exploring convertible data center network architectures with flat-tree," in [*SIGCOMM*], (2017).

[14] Chen, K., Singla, A., Singh, A., Ramachandran, K., Xu, L., Zhang, Y., Wen, X., and Chen, Y., "Osa: An optical switching architecture for data center networks with unprecedented flexibility," *IEEE/ACM Transactions on Networking* **22**, 498–511 (April 2014).

[15] Porter, G., Strong, R., Farrington, N., Forencich, A., Chen-Sun, P., Rosing, T., Fainman, Y., Papen, G., and Vahdat, A., "Integrating microsecond circuit switching into the data center," *SIGCOMM Comput. Commun. Rev.* **43**, 447–458 (Aug. 2013).

[16] Shen, Y., Rumley, S., Wen, K., Zhu, Z., Gazman, A., and Bergman, K., "Acceleration of high performance data centers using silicon photonic switch-enabled bandwidth steering," in [*Optical Communication, 2018. ECOC 2018. 44th European Conference on*], 1–2, IEEE (2018).

[17] Kim, J., Dally, W. J., Scott, S., and Abts, D., "Technology-driven, highly-scalable dragonfly topology," in [*2008 International Symposium on Computer Architecture*], 77–88 (June 2008).

[18] Kim, J., Dally, W. J., Scott, S., and Abts, D., "Cost-efficient dragonfly topology for large-scale systems," in [*2009 Conference on Optical Fiber Communication - incudes post deadline papers*], 1–3 (March 2009).

[19] "Gtc." `http://www.nersc.gov/users/computational-systems/cori/nersc-8-procurement/trinity-nersc-8-rfp/nersc-8-trinity-benchmarks/gtc/`. (Accessed on 09/27/2018).

[20] "Mpich — high-performance portable mpi." `https://www.mpich.org/`. (Accessed on 12/03/2018).