Towards All-optical Circuit-switched Datacenter Network Cores: The Case for Mitigating Traffic Skewness at the Edge

Sushovan Das
Rice University

Weitao Wang
Rice University

T. S. Eugene Ng
Rice University

ABSTRACT

All-optical circuit switched network core is the holy grail for the next-generation datacenter architectures, as electrical packet switches are struggling to cope up with increasing challenges posed by the end of Moore's law. However, traffic skewness is the biggest enemy of such all-optical network cores comprising of a simple round-robin circuit-scheduling abstraction. Even though valiant load balancing can theoretically solve the problem, it falls short in most of the practical scenarios. In this paper, we point towards a new research direction to address the skewness problem : why not resolve most of the skewness at the network edge while keeping the optical core simple? This approach is fundamentally different and can potentially enable the all-optical network core to achieve good performance in practice. We discuss relevant strategies and envision that a holistic system design is necessary considering all these strategies together.

CCS CONCEPTS

• **Networks** → **Network architectures**;

KEYWORDS

All-optical, OCS, Datacenter, Skewness

ACM Reference Format:
Sushovan Das, Weitao Wang, and T. S. Eugene Ng. 2021. Towards All-optical Circuit-switched Datacenter Network Cores: The Case for Mitigating Traffic Skewness at the Edge. In *ACM SIGCOMM 2021 Workshop on Optical Systems (OptSys'21), August 23, 2021, Virtual Event, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3473938.3474505

1 MOTIVATION

1.1 Why all-optical core is the holy grail?

As we are stepping into the post-Moore's law era, the free scaling of CMOS-based commodity Ethernet packet switches is hindered by the increasing gap between switch capacity and power/cost/latency requirements [1, 19]. As a result, the long-term sustainability of traditional packet-switched datacenter network (DCN) core is becoming more challenging. Moreover, the stringent performance requirements of diverse DCN workloads e.g., distributed deep neural network (DNN) training, high performance computing (HPC), map-reduce, frontend, database etc., makes the situation even more critical. Designing all-optical circuit-switched DCN core has gained significant attention in recent times, as optical circuit switches (OCS) have the potential to overcome the inherent challenges of electrical packet switches. Several recent proposals consider DCN architectures with all-optical circuit-switched network core [1, 12, 14]) and leverage diverse OCS technologies e.g., MEMS, AWGR etc. The fundamental advantages of OCS are **(a) agnostic to data-rate** because forwarding the photon beams does not depend on modulation rate of underlying electrical signal. **(b) negligible/zero power consumption** because of their simple operating principles e.g., mirror rotation for MEMS , diffraction for AWGR etc. **(c) no need of transceiver** because no optical-electrical-optical (O/E/O) conversion is required. **(d) negligible forwarding latency** as they do not perform packet processing and buffering. **(e) no need for frequent upgrade** because of their data-rate agnostic property and no transceiver requirement.

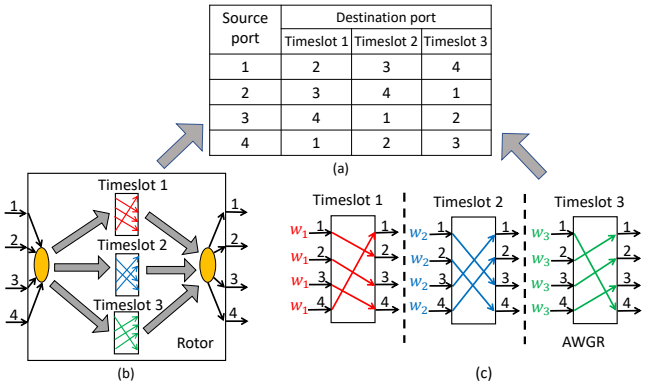


Figure 1: (a) All-optical DCN architectures share common round-robin circuit-scheduling abstraction, (b) Rotor switch realizes port-to-port mapping based on diffraction grating etched on a hard-disk drive, (c) AWGR switch realizes port-to-port mapping by wavelength routing.

1.2 Common Abstraction

In spite of using diverse underlying OCS technologies, the aforementioned DCN architectures with OCS-based core

OptSys'21, August 23, 2021, Virtual Event, USA

ACM ISBN 978-1-4503-8650-0/21/08…$15.00
https://doi.org/10.1145/3473938.3474505

share a common operational abstraction termed as round-robin circuit scheduling, as shown in figure 1(a). The OCSes are connected to a subset of end-points (i.e., ToR switches) and periodically cycle through a predefined set of circuit configurations. During a full cycle, a direct point-to-point circuit gets established once between every two end-points for equal time duration, thus providing a reconfigurably non-blocking connectivity. For example, Rotornet [12] leverages optical rotor switch consisting of micromirrors and diffraction gratings etched on a hard-disk drive. As shown in figure 1(b), each grating pattern corresponds to a *matching*, defined as a set of port-to-port circuit mapping. As the disk rotates, it cycles though a sequence of predefined matchings. Another example is Sirius [1] which leverages AWGR switches. AWGR realizes the port-to-port mapping by wavelength routing as shown in figure 1(c). A given wavelength incident to an input port gets diffracted to a unique output port creating a circuit. During a timeslot, a particular wavelength is assigned to all the input ports leading to a logical circuit-schedule. Eventually, a sequence of different wavelength assignment completes a cycle. Even non-optical circuit-switched network design such as Shoal [17] adopts this abstraction to provide intra-rack connectivity for disaggregated setting. This abstraction has several practical advantages as follows.

(1) Very simple to operate such coordination-free or open-loop switching. As the OCS goes through a fixed set of circuit patterns repeatedly, no on-demand network-wide synchronization is required.
(2) More scalable in practice because there is no need of centralized scheduler to gather the network-wide traffic matrix and calculate the circuit assignments.
(3) Achieve 100% throughput for uniform traffic in theory, because every end-point pair gets equal bandwidth connectivity within one cycle.

1.3 Traffic skewness is the reality

However, several recent works have observed that realistic DCN traffic is not uniform, rather heavily skewed i.e., a small subset of source-destination pairs exchange a significant fraction of traffic while most of the pairs have almost no traffic. For example, analysis of Microsoft DCN trace reveals that 80% of traffic is exchanged between $0.03 - 0.4\%$ of hot-rack pairs [7, 10]. Traffic traces of emerging disaggregated workloads consisting of various applications (such as interactive queries, graph processing etc.) show heavy skewness as well, where 33% of the nodes generate more than 84% of the flows [6, 17]. Besides, frontend trace from Facebook DCN also shows highly skewed inter-rack traffic pattern [11, 16, 23]. Skewed access-popularity across objects leads to such imbalance in the cache cluster [8].

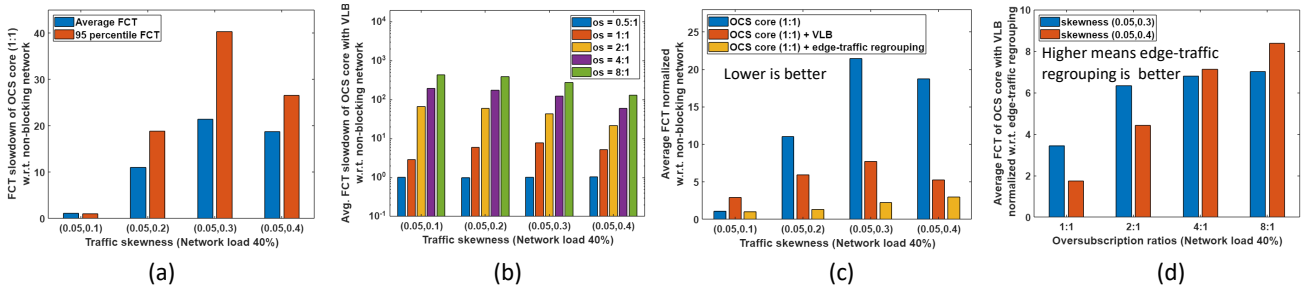
1.4 Skewness: Nemesis of round-robin circuit scheduling abstraction

Intuitively, traffic skewness is the enemy of round-robin circuit scheduling abstraction because bandwidth among all the source-destination pairs are uniformly distributed. As a result, circuits between the hot end-point pairs are heavily utilized, while the abstraction cannot leverage underutilized bandwidth of the cold circuits. To quantify the problem, we compare the performance of a round-robin all-optical circuit-switched core having $1:1$ ToR uplink bandwidth (i.e., aggregated uplink bandwidth is the same as aggregated downlink bandwidth) with an ideal non-blocking network. We perform simulations extending a packet-level simulator [15] which supports TCP transport. Both architectures consist of 16 ToR switches and 32 servers per ToR. We generate flow-level cache traffic trace having the inter-arrival time and flow size distribution obtained from [16]. On top of that, we introduce traffic skewness based on a simple model. We define skewness parameter (x, y) where x fraction of hot-rack pairs exchange y fraction of the traffic. The remaining traffic is uniformly distributed among other rack pairs. Figure 2(a) shows average and 95 percentile flow completion time (FCT) slowdown for round-robin OCS-based $(1:1)$ network compared to an ideal non-blocking network. Overall, our simulation validates that round-robin OCS-based $(1:1)$ network performs close to non-blocking network when the traffic is almost uniform (skewness $(0.05, 0.1)$). However, its performance degrades significantly (average and 95 percentile FCT slowdown are upto $18.7\times$ and $40.3\times$ respectively) with higher traffic skewness. Note that, relative FCT slowdown at skewness $(0.05, 0.4)$ is little smaller than at skewness $(0.05, 0.3)$ because non-blocking network performance degrades at a slightly higher rate than round-robin OCS-based $(1:1)$ network.

1.5 Is valiant load balancing enough?

Valiant load balancing (VLB) [22] is the state-of-the-art technique to improve network throughput in presence of skewed traffic. In fact, all these round-robin all-optical [1, 12, 14] and non-optical [17] circuit-switched network design proposals consider VLB as the most promising approach to deal with traffic skewness. Besides direct one-hop path, VLB also leverages two-hop indirect path between any source-destination pair. If there is no direct circuit between a pair of end-points at current time slot, VLB allows the source to forward the traffic to an intermediate end-point (source having a direct circuit currently). The intermediate end-point can forward the traffic to destination during a later time slot. Although VLB improves the throughput compared to bare round-robin scheduling, it poses several challenges as follows.

(1) Theoretically, a round-robin OCS-based network core enabled with two-hop VLB would be able to use only 50% of the core bandwidth in worst case.
(2) The intermediate end-point needs to buffer data until it gets a direct circuit with the destination. Such buffering can significantly degrade the FCT for small flows.
(3) The all-optical network design proposals deploy customized congestion control mechanism to avoid the



(a) (b) (c) (d)

Figure 2: (a) Average and 95 percentile FCT slowdown of round-robin OCS-based (1 : 1) network core w.r.t. non-blocking network vs. traffic skewness, (b) Average FCT slowdown of OCS-based network core enabled with valiant load balancing w.r.t. non-blocking network vs. traffic skewness, at different oversubscription scenarios, (c) Average FCT normalized w.r.t. non-blocking network vs. traffic skewness for different OCS-based network core (1 : 1) scenarios (lower is better), (d) Average FCT of VLB-enabled OCS-based network core normalized w.r.t. edge-traffic regrouping vs. oversubscription ratio, at different traffic skewness (higher means edge-traffic regrouping is better).

buffer overflow. It assumes that each end-point maintains a separate FIFO queue for every other end-point and eventually bounds the queue length. Maintaining a separate intermediate queue for every potential destination is not a scalable solution.

(4) Employing all these techniques together significantly increases the system complexity and defeats the purpose of a simple scheduler-less optical network design.

Figure 2(b) shows the relative slowdown of average FCT for an OCS-based circuit-switched network enabled with VLB compared to non-blocking network, at different oversubscription (os) scenarios. Our simulation validates the fact that VLB can perform very close to a non-blocking network if the core bandwidth is doubled (i.e., os 0.5 : 1). However, doubling the core bandwidth statically commits to the worst case traffic which may not appear all the time but if appears, cannot be ignored. As a result, the strategy can be wasteful if the traffic is close to uniform or the network is not heavily loaded. Moreover, the network core can be highly oversubscribed in reality [16, 19]. Based on our observation, VLB improves the performance compared to bare round-robin scheduling when traffic skewness is high, but the performance of VLB enabled OCS-based core is far from ideal non-blocking network. Even for os 1 : 1, VLB atop round-robin OCS-based core (1 : 1) performs $2.9 - 7.7\times$ worse than a non-blocking network. Additionally, the performance degrades rapidly with higher oversubscriptions. At skewness $(0.05, 0.4)$, the average FCT slowdown are $21.8\times$, $59.3\times$ and $130.8\times$ for os 2 : 1, os 4 : 1 and os 8 : 1 respectively.

2 INSIGHT

We believe that OCS-based DCN core designs can go a long way if the network can efficiently handle the skewed traffic pattern without over-provisioning the core bandwidth. Driven by the observations, we envision a fundamentally different insight: regroup the edge traffic intelligently so that most of the traffic skewness is resolved at the network edge and the remaining traffic at network core becomes almost uniform, which is already favorable scenario for simple round-robin circuit scheduling abstraction. Two major aspects of intelligent edge-traffic regrouping are as follows.

(1) **Localize the traffic within a rack whenever possible.** Converting the inter-rack traffic to an intra-rack traffic can reduce the heavy utilization of hot-circuits and mitigate the congestion. This way, it can significantly alleviate the skewness at network core and frees up some core network bandwidth which can be provisioned for future traffic demands.

(2) **Load balance the traffic across the uplinks of different racks.** If traffic localization is not possible, then uniformly distributing the traffic across the racks can mitigate the traffic imbalance at the core. This in turn leads to near-uniform utilization of the circuits, making the core network traffic almost uniform.

We use a greedy heuristic to understand the potential benefit of intelligent edge-traffic regrouping. Our heuristic tries its best to localize most of the traffic within racks. Then it distributes the remaining traffic across the uplinks of different racks as uniformly as possible. During the simulation, we perform this heuristic at every 1 second interval. Figure 2(c) shows that intelligent edge-traffic regrouping on OCS-based network core (1 : 1) can potentially mitigate the skewness to a great extent and performs closest to the non-blocking network compared to the state-of-the-art approaches. Figure 2(d) shows that edge-traffic regrouping significantly outperforms VLB across different oversubscription scenarios (average FCT slowdown of VLB is $1.7 - 8.3\times$ compared to edge-traffic regrouping) irrespective of traffic skewness.

3 POTENTIAL STRATEGIES

3.1 Definition

There are three potential strategies that can possibly contribute in regrouping the traffic at the edge. Figure 3 shows the examples of all three strategies together.

(a) **Job placement** (figure 3(a)) refers to place a distributed application intelligently across a set of hosts before the application starts execution [2–5, 9, 13, 21, 24, 25].

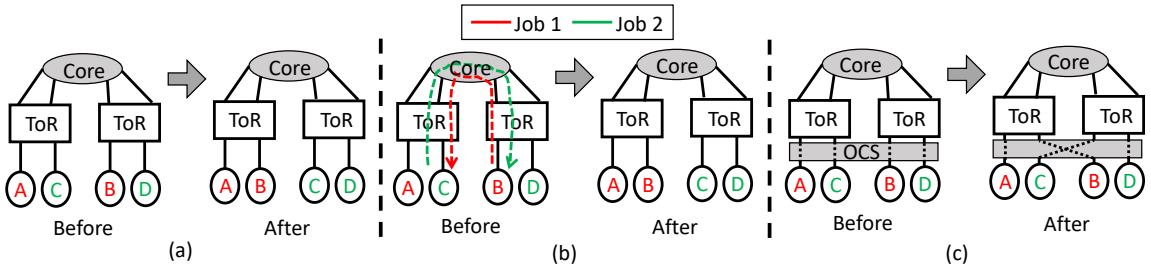


Figure 3: Examples of (a) job placement, (b) job migration, (c) dynamic edge topology reconfiguration.

However, job placement allows one-shot decision, therefore cannot be changed during application runtime.

(b) **Job migration** (figure 3(b)) refers to migrate an application at runtime, from one host to another [18, 20]. However, downside of job migration is that it injects extra traffic as application state needs to be copied across the network.

(c) **Dynamic edge topology reconfiguration** (figure 3(c)) removes the logical rack boundary. It leverages reconfigurable OCS and enables a host to move across a subset of ToR switches during application runtime [23]. This is equivalent to migrate a job without injecting migration traffic and the runtime overhead depends on OCS reconfiguration delay.

3.2 Pros and cons of these strategies

We intuitively examine the pros and cons of all three strategies based on the relevant application-level attributes such as job duration (total runtime), job resource usage (number of hosts), job size (state memory per host) and job traffic pattern variability.

Job placement:

(1) Job placement is well-suited for the jobs consuming less resource because the job is more likely to be localized even if the resource is fragmented.

(2) Applications with one traffic pattern, i.e., no variability, can get complete benefit from job placement strategy.

(3) Finding optimal placement for large resource consuming jobs can be hard, as the number of potential choices will be limited due to resource fragmentation.

(4) Job placement is not favorable to the applications having predictable but varying traffic pattern, because it only allows one-shot decision. In most cases, it is impossible to find a placement that can optimize all different traffic patterns simultaneously.

(5) Job placement decision cannot be modified if the available bandwidth changes among end-points at runtime.

Job migration:

(1) Job migration is ideal for the long duration jobs having small per-host state memory, because the injected migration traffic volume is low and migration time is a small fraction of overall runtime.

(2) Job migration can easily optimize the applications with variable traffic pattern, as migration can be performed during application runtime.

(3) Job can be migrated dynamically, if available bandwidth between two end-points changes during runtime.

(4) Job migration can potentially reduce the resource fragmentation for large resource consuming jobs.

(5) Job migration is heavily disruptive for short duration jobs leading to high runtime overhead.

(6) Job migration can be even worse for the applications having large state memory, because it will inject heavy migration traffic contributing to network congestion. For similar reason, job migration is not well-suited under heavy network load.

Dynamic edge topology reconfiguration:

(1) Dynamic edge topology reconfiguration is well suited for long duration jobs because of low runtime overhead. Emerging OCS technologies lead to smaller reconfiguration delay reducing the overhead further.

(2) Dynamic edge topology reconfiguration can adapt with variable application-level traffic pattern and dynamic bandwidth change between end-points.

(3) Dynamic edge topology reconfiguration is independent of job size because it logically moves the host instead of injecting any migration traffic to the network. Due to the same reason, it is beneficial for the applications irrespective of the network load.

(4) Dynamic edge topology reconfiguration is not well-suited for jobs having runtime of the same order as OCS reconfiguration time.

4 CONCLUSION: A HOLISTIC VIEW

Existing works have treated these strategies independently, not in a holistic manner. After examining these strategies carefully, we recognize that each of them are powerful tool but may not be sufficient alone to mitigate the skewness completely at different scenarios. Moreover, these strategies are correlated. For example a job placement, which is optimal for the current edge topology, can be very sub-optimal after the edge topology reconfiguration. Therefore, on one hand, potential benefit for the applications can improve if these strategies are leveraged in a cooperative manner. On the other hand, non-cooperative or conflicting usage of these strategies can nullify the benefit in reality. Therefore, we need to envision a system that leverage all three strategies holistically and make them cooperate whenever necessary. Such holistic system design can make all-optical circuit-switched network cores widely acceptable and adoptable to the community.

5 ACKNOWLEDGEMENT

We thank the anonymous reviewers for their insightful feedback. This research is sponsored by the NSF under CNS-1718980, CNS-1801884, and CNS-1815525.

REFERENCES

[1] Hitesh Ballani, Paolo Costa, Raphael Behrendt, Daniel Cletheroe, Istvan Haller, Krzysztof Jozwik, Fotini Karinou, Sophie Lange, Kai Shi, Benn Thomsen, et al. 2020. Sirius: A flat datacenter network with nanosecond optical switching. In *Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication*. 782–797.

[2] Sergey Blagodurov, Alexandra Fedorova, Evgeny Vinnik, Tyler Dwyer, and Fabien Hermenier. 2015. Multi-objective job placement in clusters. In *SC'15: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE, 1–12.

[3] Peter Bodík et al. 2012. Surviving failures in bandwidth-constrained datacenters. In *SIGCOMM*. ACM.

[4] Mosharaf Chowdhury, Srikanth Kandula, and Ion Stoica. 2013. Leveraging endpoint flexibility in data-intensive clusters. In *ACM SIGCOMM Computer Communication Review*, Vol. 43. ACM, 231–242.

[5] Mosharaf Chowdhury, Matei Zaharia, Justin Ma, Michael I Jordan, and Ion Stoica. 2011. Managing data transfers in computer clusters with orchestra. *ACM SIGCOMM Computer Communication Review* 41, 4 (2011), 98–109.

[6] Peter X Gao, Akshay Narayan, Sagar Karandikar, Joao Carreira, Sangjin Han, Rachit Agarwal, Sylvia Ratnasamy, and Scott Shenker. 2016. Network requirements for resource disaggregation. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*. 249–264.

[7] Monia Ghobadi, Ratul Mahajan, Amar Phanishayee, Nikhil Devanur, Janardhan Kulkarni, Gireeja Ranade, Pierre-Alexandre Blanche, Houman Rastegarfar, Madeleine Glick, and Daniel Kilper. 2016. Projector: Agile reconfigurable data center interconnect. In *Proceedings of the 2016 ACM SIGCOMM Conference*. 216–229.

[8] Qi Huang, Helga Gudmundsdottir, Ymir Vigfusson, Daniel A Freedman, Ken Birman, and Robbert van Renesse. 2014. Characterizing load imbalance in real-world networked caches. In *Proceedings of the 13th ACM Workshop on Hot Topics in Networks*. 1–7.

[9] Virajith Jalaparti et al. 2015. Network-aware scheduling for data-parallel jobs: Plan when you can. *SIGCOMM* (2015).

[10] Simon Kassing, Asaf Valadarsky, Gal Shahaf, Michael Schapira, and Ankit Singla. 2017. Beyond fat-trees without antennae, mirrors, and disco-balls. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*. 281–294.

[11] Zaoxing Liu, Zhihao Bai, Zhenming Liu, Xiaozhou Li, Changhoon Kim, Vladimir Braverman, Xin Jin, and Ion Stoica. 2019. Distcache: Provable load balancing for large-scale storage systems with distributed caching. In *17th USENIX Conference on File and Storage Technologies (FAST 19)*. 143–157.

[12] William M Mellette, Rob McGuinness, Arjun Roy, Alex Forencich, George Papen, Alex C Snoeren, and George Porter. 2017. Rotornet: A scalable, low-complexity, optical datacenter network. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*. 267–280.

[13] Xiaoqiao Meng, Vasileios Pappas, and Li Zhang. 2010. Improving the scalability of data center networks with traffic-aware virtual machine placement. In *2010 Proceedings IEEE INFOCOM*. IEEE, 1–9.

[14] George Porter, Richard Strong, Nathan Farrington, Alex Forencich, Pang Chen-Sun, Tajana Rosing, Yeshaiahu Fainman, George Papen, and Amin Vahdat. 2013. Integrating microsecond circuit switching into the data center. *ACM SIGCOMM Computer Communication Review* 43, 4 (2013), 447–458.

[15] Costin Raiciu, Christoph Paasch, Sebastien Barre, Alan Ford, Michio Honda, Fabien Duchene, Olivier Bonaventure, and Mark Handley. 2012. How hard can it be? designing and implementing a deployable multipath TCP. In *9th USENIX symposium on networked systems design and implementation (NSDI 12)*. 399–412.

[16] Arjun Roy, Hongyi Zeng, Jasmeet Bagga, George Porter, and Alex C Snoeren. 2015. Inside the social network's (datacenter) network. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*. 123–137.

[17] Vishal Shrivastav, Asaf Valadarsky, Hitesh Ballani, Paolo Costa, Ki Suh Lee, Han Wang, Rachit Agarwal, and Hakim Weatherspoon. 2019. Shoal: A network architecture for disaggregated racks. In *16th USENIX Symposium on Networked Systems Design and Implementation (NSDI 19)*. 255–270.

[18] Vivek Shrivastava, Petros Zerfos, Kang-Won Lee, Hani Jamjoom, Yew-Huey Liu, and Suman Banerjee. 2011. Application-aware virtual machine migration in data centers. In *2011 Proceedings IEEE INFOCOM*. IEEE, 66–70.

[19] Arjun Singh, Joon Ong, Amit Agarwal, Glen Anderson, Ashby Armistead, Roy Bannon, Seb Boving, Gaurav Desai, Bob Felderman, Paulie Germano, et al. 2015. Jupiter rising: A decade of clos topologies and centralized control in google's datacenter network. *ACM SIGCOMM computer communication review* 45, 4 (2015), 183–197.

[20] Georg Stellner. 1996. CoCheck: Checkpointing and process migration for MPI. In *Proceedings of International Conference on Parallel Processing*. IEEE, 526–531.

[21] Xiongchao Tang, Haojie Wang, Xiaosong Ma, Nosayba El-Sayed, Jidong Zhai, Wenguang Chen, and Ashraf Aboulnaga. 2019. Spread-n-share: improving application performance and cluster throughput with resource-aware job placement. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. 1–15.

[22] Leslie G. Valiant. 1982. A scheme for fast parallel communication. *SIAM journal on computing* 11, 2 (1982), 350–361.

[23] Dingming Wu, Weitao Wang, Ang Chen, and TS Eugene Ng. 2019. Say no to rack boundaries: Towards a reconfigurable pod-centric dcn architecture. In *Proceedings of the 2019 ACM Symposium on SDN Research*. 112–118.

[24] Matei Zaharia, Dhruba Borthakur, Joydeep Sen Sarma, Khaled Elmeleegy, Scott Shenker, and Ion Stoica. 2010. Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling. In *Proceedings of the 5th European conference on Computer systems*. 265–278.

[25] Christopher Zimmer, Saurabh Gupta, Scott Atchley, Sudharshan S Vazhkudai, and Carl Albing. 2016. A multi-faceted approach to job placement for improved performance on extreme-scale systems. In *SC'16: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE, 1015–1025.