Rearchitecting Datacenter Networks: A New Paradigm with Optical Core and Optical Edge

Sushovan Das, Arlei Silva, T. S. Eugene Ng
Rice University

***Abstract*—All-optical circuit-switching (OCS) technology is the key to design energy-efficient and high-performance datacenter network (DCN) architectures for the future. However, existing round-robin based OCS cores perform poorly under realistic workloads having high traffic skewness and high volume of inter-rack traffic. To address this issue, we propose a novel DCN architecture *OSSV*: a combination of OCS-based core (between ToR switches) and OCS-based reconfigurable edge (between servers and ToR switches). On one hand, the OCS core is traffic agnostic and realizes reconfigurably non-blocking ToR-level connectivity. On the other hand, OCS-based edge reconfigures itself to reshape the incoming traffic in order to jointly minimize traffic skewness and inter-rack traffic volume. Our novel optimization framework can obtain the right balance between these intertwined objectives. Our extensive simulations and testbed evaluation show that OSSV can achieve high performance under diverse DCN traffic while consuming low power and incurring low cost.**

***Index Terms*—All-optical, Datacenter Network, Architecture**

I. Introduction

Datacenter Networks (DCN) are the key infrastructure for diverse applications such as iterative machine learning (ML), distributed deep neural network training (DNN), high performance computing (HPC), frontend, database, etc., having stringent performance requirements. Moreover, the popularity of domain-specific accelerators (GPUs, TPUs) and non-volatile memories (NVM) is further shifting the major bottleneck from computation to network IO [41]. However, traditional packet-switched network cores in today's DCNs are not sustainable in the long run as CMOS-based electrical packet switches face the challenge posed by the end of Moore's Law [12], [38]. Fundamentally, the commodity Ethernet switches are excessively power-hungry and it escalates at a faster rate compared to the switching capacity, thus hindering the free scaling for next-generation clusters. For example, Broadcom's Tomahawk-4 64×400 Gbps, the fastest Ethernet switch ASIC commercially available on the market today, has a power consumption of around 2.5 KW [5]. A non-blocking network topology consisting of such high-speed packet switches would consume prohibitively high power that can go beyond the infrastructure energy budget [12], leading to high carbon footprint [32]. Under such energy-critical situations, optical circuit-switching (OCS) technology having fundamental properties such as a) agnostic to data rate, b) negligible power consumption, c) negligible forwarding latency, etc., seems to be the most promising alternative. This in turn fuels the necessity to envision the all-optical circuit-switched cores [12], [29], [30], [34] for designing the sustainable, energy-efficient and high-performance future-generation DCN architectures.

Despite of leveraging diverse OCS technologies such as MEMS [30], [34], AWGR [12] etc., the existing OCS core-based DCN architectures share a common operational abstraction termed as round-robin circuit scheduling to achieve reconfigurably non-blocking connectivity among ToR switches. However, this operational abstraction is fundamentally incompatible with diverse DCN workloads. More specifically, the performance of such OCS cores heavily suffers if there is a) high traffic skewness i.e., a small subset of source-destination pairs exchange a significant amount of traffic [20], [21], [23], [25], [28], [36], [41], and b) high inter-rack traffic volume [13]–[15], [36]. The abstraction provides uniform bandwidth distribution among ToR switches over time. Hence, the circuits between hot rack-pairs are heavily utilized under skewed traffic, while it cannot leverage the underutilized bandwidth of the cold circuits. The situation can worsen if the core is over-subscribed, as the high volume of skewed inter-rack traffic would contend for bandwidth and face severe congestion.

Driven by such observation, we envision a fundamentally different approach: regroup the edge traffic intelligently so that a) most of the traffic gets localized within a rack reducing the inter-rack traffic volume, and b) the remaining inter-rack traffic at the network core becomes almost uniform reducing the traffic skewness. However, jointly minimizing the traffic skewness and inter-rack traffic volume is a non-trivial problem, as these objectives are intertwined. In most cases, blindly optimizing for one of the objectives can adversely affect the other, leading to significant performance gap between OCS-based cores and non-blocking packet-switched cores. For instance, skewness minimization could spread the traffic across racks, having zero or very little traffic localization. But aggressive traffic localization could reduce inter-rack traffic volume significantly while the remaining inter-rack traffic would be highly skewed. Hence, jointly optimizing these objectives while obtaining the right balance is fundamentally challenging.

In this paper, we propose a novel DCN architecture *OSSV* (Optical Substrate for Skewness and inter-rack traffic Volume minimization): a combination of a round-robin optical core and a reconfigurable optical edge that can efficiently handle diverse DCN workloads. To realize the reconfigurable edge, we place optical circuit switches between servers and ToRs that provides the flexibility to move the servers across the racks. On one hand, the OCS-based round-robin core can be traffic agnostic and provide reconfigurably non-blocking connectivity among ToRs. On the other hand, OCS-based edge can dynamically reconfigure itself in order to jointly minimize

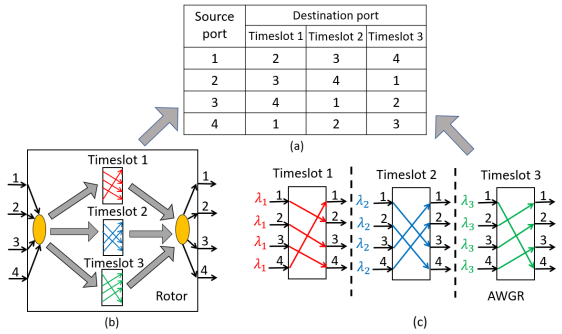

Figure 1: (a) All-optical DCN architectures share common round-robin circuit scheduling abstraction [17], (b) Rotor switch realizes port-to-port mapping based on diffraction grating etched on a hard-disk drive, (c) AWGR switch realizes port-to-port mapping by wavelength routing.

the traffic skewness and inter-rack traffic volume. This way, the optical edge reshapes the incoming traffic so that most of the traffic avoids the core bandwidth contention and the remaining inter-rack traffic is almost uniform, a favorable scenario for the round-robin cores. We design a novel optimization framework for joint Skewness and inter-rack traffic Volume (SV) minimization, that can obtain the right balance between these intertwined objectives. Our extensive simulations and testbed evaluation show that OSSV, equipped with novel SV-minimization framework, can significantly reduce the performance gap between all-optical cores vs. non-blocking packet-switched networks, under diverse DCN traffic. Our detailed power and cost analyses reveal that OSSV can be up to $2.1\times$ more power-efficient and $1.28\times$ more cost-efficient, compared to an OCS-based non-blocking network core.

II. Motivation

A. Why all-optical core is promising?

Designing OCS-based DCN cores, such as RotorNet, Sirius, etc. [12], [30], [34], has attracted significant interest in recent times, as OCS has several fundamental advantages over packet switches discussed as follows.

1) **Agnostic to data-rate** because forwarding the photon beams does not depend on modulation rate of underlying electrical signal;
2) **Negligible/zero power consumption** because of their simple operating principles e.g., mirror rotation for MEMS-based OCS (negligible power required), diffraction for AWGR-based OCS (fully passive) etc;
3) **No need of transceiver** because no optical-electrical-optical (O/E/O) conversion is required at the core;
4) **Negligible forwarding latency** as they do not perform packet processing and buffering;
5) **No need for frequent upgrade** because of their data-rate agnostic property and no transceiver requirement.

B. Common abstraction

In spite of using diverse underlying OCS technologies, the aforementioned DCN architectures share a common operational abstraction termed as round-robin circuit scheduling (Figure 1(a)). The OCSes are connected to a subset of 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 ToR switches for equal time duration, thus providing a reconfigurably non-blocking connectivity [17]. For example, RotorNet [30] leverages an optical rotor switch consisting of micromirrors and diffraction gratings etched on a hard-disk drive. Each grating pattern corresponds to a *matching*, defined as a set of port-to-port circuit mapping (Figure 1(b)). As the disk rotates, it cycles through a sequence of predefined matchings. Another example is Sirius [12] which leverages AWGR switch-based wavelength routing to realize the port-to-port mapping (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 each input port leading to a logical circuit-schedule. Eventually, a sequence of different wavelength assignments completes a cycle.

C. Poor performance under diverse DCN workloads

The round-robin OCS-based cores are fundamentally incompatible with diverse DCN workloads where the traffic is a) highly skewed (i.e., a small subset of source-destination pairs exchange a significant fraction of the traffic while most of the pairs have almost no traffic), and b) heavily inter-rack due to large-scale deployment [9], resource fragmentation [22], specific placement constraints [1], etc. For example, analysis of Microsoft DCN trace reveals that 80% of traffic is exchanged between $0.03 - 0.4\%$ of hot-rack pairs [21], [25]. Traffic traces of emerging disaggregated workloads (e.g., interactive queries, graph processing etc.) show heavy skewness as well, where 33% of the nodes generate more than 84% of the flows [20], [37]. Besides, a frontend trace from Facebook DCN also shows highly skewed inter-rack traffic pattern [28], [36], [41]. Skewed access-popularity across objects leads to such imbalance in the cache cluster [23]. Moreover, a frontend trace collected from Facebook [36] has 96.26% inter-rack traffic [36], [41]. Similar trends exist for database and Hadoop traces, having 92.89% and 52.49% inter-rack traffic, respectively [36].

Intuitively, high degree of rack-level traffic skewness is the enemy of such round-robin OCS-based cores because bandwidth among all the rack pairs is uniformly distributed. As a result, it can achieve 100% throughput only under uniform traffic. However, in the presence of skewed traffic, circuits between the hot rack pairs are heavily utilized, while the abstraction cannot leverage underutilized bandwidth of the cold circuits. High inter-rack traffic volume of DCN workloads makes the situation even worse if the core is oversubscribed.

To quantify this issue, we compare the performance of a round-robin OCS-based core with an ideal non-blocking network, in the presence of heavily inter-rack traffic, while varying traffic skewness and oversubscription (os) at the core. We perform simulations extending a packet-level simulator [35] which supports TCP transport. Both architectures consist of 16 ToR switches and 32 servers per ToR and 100 Gbps transmission speed. The OCS core is emulating Sirius [12],

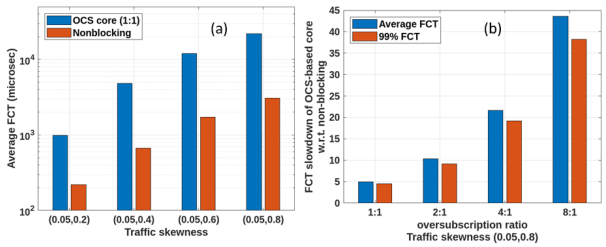

Figure 2: (a) Performance of a round-robin OCS core degrades with high traffic skewness, (b) Performance gap between OCS core and a non-blocking network core significantly increases under high oversubscription, due to heavy inter-rack traffic.

with 100 nsec OCS downtime and 99% duty-cycle. We generate flow-level cache traffic traces having inter-rack traffic volume, inter-arrival time and flow size distribution obtained from [36]. 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 the performance of a round-robin OCS core (os $1:1$) and a non-blocking network while varying the traffic skewness. The average flow completion time (FCT) slowdown of the OCS core can be up to $7.2\times$. The performance degrades rapidly with a higher os ratio due to heavy inter-rack traffic volume (Figure 2(b)). For example, at $8:1$ os, the average FCT slowdown can be more than $43\times$.

III. Our Approach: Edge Traffic Regrouping

A. Primary insight

Driven by the previous observations, we envision a fundamentally different approach: regroup the edge traffic intelligently with the following goals.

1) **Minimization of inter-rack traffic volume.** Converting the inter-rack traffic to intra-rack traffic can reduce the heavy utilization of hot-circuits and mitigate the congestion. Thus, it frees up some core network bandwidth which can be provisioned for future traffic demands.
2) **Minimization of inter-rack traffic skewness.** If traffic localization is not possible, then uniformly distributing the traffic across the racks can mitigate traffic imbalance at the core. This leads to near-uniform utilization of the circuits, making the core network traffic almost uniform.

B. Skewness minimization & traffic localization: Intertwined

Minimizing traffic skewness and improving traffic locality are not independent objectives, and naïvely optimizing for one can adversely affect the other in several scenarios. Ideally, if it is possible to localize all the network traffic under racks (i.e., inter-rack traffic volume is zero), then only traffic localization can completely mitigate the traffic skewness. However, skewness minimization can also be achieved by spreading all the traffic across racks, having zero traffic localization. Such an allocation would suffer from severe performance degradation if the core has bandwidth oversubscription. Also, an aggressive traffic localization strategy can lead to a scenario where the remaining inter-rack traffic could be heavily skewed, harming the overall fairness. Hence, jointly minimizing traffic skewness and inter-rack traffic volume while obtaining the right balance is non-trivial and fundamentally challenging. In Section V, we explain this problem in more detail and develop novel optimization framework for joint Skewness and inter-rack traffic Volume (SV) minimization.

C. Our strategy: Dynamic edge topology reconfiguration

We envision dynamic edge topology reconfiguration as the most promising strategy for edge traffic regrouping. It leverages reconfigurable OCSes at the edge (i.e., between server pools and ToRs) and enables a host to move across a subset of ToR ports during runtime, effectively removing the logical rack boundary. First, introducing reconfigurability at the optical edge is unique, logically closer to the application, and complementary to the all-optical network cores. In general, the edge reconfigurability provides a wide range of choices e.g., localize a group of servers under a ToR, spread those servers across different ToRs, realize dynamic ToR-to-ToR topology slices etc., based on diverse optimization objectives. This paper specifically focuses on reconfigurable edge design to reduce traffic skewness and inter-rack traffic volume. Second, the reconfigurable optical edge can move the servers across racks at the cost of a small OCS reconfiguration downtime which is independent of job size and beneficial to applications irrespective of network load and core bandwidth oversubscription. Such ability can rectify the impact of sub-optimal application placements without modifying the ToR-to-ToR circuit schedule at the network core. Also, the emerging OCS technologies (e.g., 2D-MEMS, AWGR etc.) with smaller reconfiguration downtime would reduce the reconfiguration overhead further, making our approach even more attractive. Third, this approach improves the dataplane reliability as the server-to-ToR regrouping strategies enable unique ways to recover from ToR-to-ToR link failure. Additionally, such an edge decouples the server from a particular ToR switch, thus enabling a smooth recovery under a single server-to-ToR link failure assuming each ToR has some "free" backup ports [42]. Understanding the impact of failure and adapting different strategies can be future work.

IV. Proposed Architecture: OSSV

A. Connectivity structure

In this section, we concretely discuss our proposed architecture OSSV (Optical Substrate for Skewness and inter-rack traffic Volume minimization) which has two components: a) a traffic-agnostic round-robin OCS-based core to realize reconfigurably non-blocking connectivity across ToRs, and b) an OCS-based reconfigurable edge (between servers and ToRs) that adapts to different traffic patterns in order to jointly minimize the traffic skewness and inter-rack traffic volume (SV Minimization). Figure 3 shows an example OSSV connectivity structure. Ideally, a single big port-count edge-OCS spanning the whole datacenter would provide maximum flexibility for a server to move across any racks. However, such a design would be impractical and not scalable as, a) the commercial OCS port count is too small to span across a

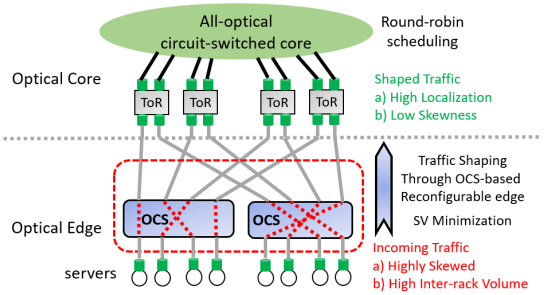

Figure 3: OSSV: a combination of OCS-based round-robin core and OCS-based reconfigurable edge.

datacenter, and b) a centralized datacenter-wide control plane would be necessary to execute traffic-aware SV minimization which is hard to scale. Hence, we consider a scalable design with multiple edge-OCS (Figure 3) where a server can move across a subset of rack ports, decoupling our architecture from specific OCS technology, port count limitations and datacenter-wide control. Additionally, such a design mitigates a single point of failure at the edge. For example, all-optical core with 16 ToRs and 32 servers per ToR can have 8 edge-OCS with 128 ports each. Each OCS connects with 4 ports from each ToR. Therefore, servers connected to that OCS can only move across those designated ToR ports.

B. Realization of OSSV edge

Potentially we can realize the OSSV edge by inserting any circuit-switching technology. However, we specifically focus on optical circuit switches (OCS) e.g., MEMS, AWGR etc., because of their inherent advantages (Section II-A). Only the mechanism of runtime circuit reconfiguration would vary based on the underlying OCS technology. For MEMS-based OCS, the new port-to-port mapping request is directly sent to the OCS. For AWGR switches, the circuit reconfiguration can be realized by changing the wavelength of tunable optical transceivers at the servers/ToR downlinks associated with the given OCS. Another technology-specific parameter is the reconfiguration downtime, which can be of the order of millisec (3D-MEMS), microsec (2D-MEMS), and nanosec (AWGR).

C. OSSV control plane

We envision a distributed control plane for OSSV, where each edge-OCS is equipped with a controller and reconfigured based on its local traffic, therefore not requiring network-wide traffic information. Each OSSV controller sends the reconfiguration command to the designated OCS and new routing rules to the designated ToR ports. The suitable reconfiguration epoch depends on the workload, and can be operator-tuned.

The workflow of the distributed OSSV control plane is as follows: Each OSSV controller (1) reactively/passively monitors and collects traffic statistics by querying the flow counters at the designated ToR ports; (2) performs SV minimization (Section V) and determines the optimal edge topology (i.e., server-to-ToR mapping); (3) generates new routes and install them on ToR switches; and (4) sends circuit-reconfiguration requests to OCS and activate those new routing rules on packet switches simultaneously. Only during the OCS reconfiguration in step (4), the flows face the physical downtime.

To collect the traffic statistics, the OSSV controller relies on periodic flow counters from the ToRs. Based on the analysis and measurements provided in recent works [11], [31], [36], [41], such a scheme is scalable as, a) memory constraints in the current packet switches are being relaxed with higher SRAM capacity and b) concurrent flows per servers are being bounded (within a thousand). The OSSV controller relies on traffic stability and estimates the true traffic demand using a fast-converging heuristic like Hedera [10]. In summary, Hedera identifies the elastic flows (i.e., flows bottlenecked by the network, not by the application) and computes their ideal max-min bandwidth share, thus mitigating the observation bias based on the current topology. Finally, those estimated flow demands are aggregated into a server-to-server traffic matrix and become the input of the SV minimization algorithm.

We envision epoch-based operation of OSSV control plane, where the frequency of pulling the traffic statistics by the controller and optimizing the edge-topology is the primary design parameter that impacts the final performance. Intuitively, if the reconfiguration epoch is too long, it leads to sub-optimal performance due to coarse-grained monitoring. As a result, the derived topology will be far from optimal, as the network condition has changed significantly by then. Also, if the epoch interval is too short, it will penalize the performance due to several unnecessary reconfiguration downtimes. Based on our simulation, there is a sweet spot in terms of reconfiguration epoch interval, achieving the best trade-off between controller observation frequency and OCS reconfiguration penalty. The optimal epoch interval of a given workload is correlated with temporal traffic stability and average flow duration under full-bisection bandwidth DCN, which can be pre-computed based on historical flow-level statistics. We quantify the performance impact of reconfiguration epoch interval in Section VI-E.

To design the OSSV control plane, one interesting research question we address is: How the OCS port count affects the interplay between a) traffic statistics collection and topology optimization overhead, and b) end performance. Intuitively with smaller OCS port count, the size of traffic matrix will decrease which makes the SV minimization perform faster. However, the flexibility of moving a server across different ToR ports will be constrained by the OCS port count, which may lead to performance degradation. In sections VI-F and VI-G, we study the interplay of these two parameters with different OCS port counts. Based on our simulation, under highly skewed cache traffic on a round-robin OCS core with os ratio $8:1$ and 512 servers, 8 edge-OCS (128 ports) achieves $6.6\times$ computation speedup compared to 1 edge-OCS (1024 ports) while having 20.5% degradation in average FCT.

V. SV Minimization Framework

A. Generalized problem formulation

The goal of SV minimization is to jointly minimize the traffic skewness and inter-rack traffic volume. As discussed in Section III-B, these two objectives are intertwined which

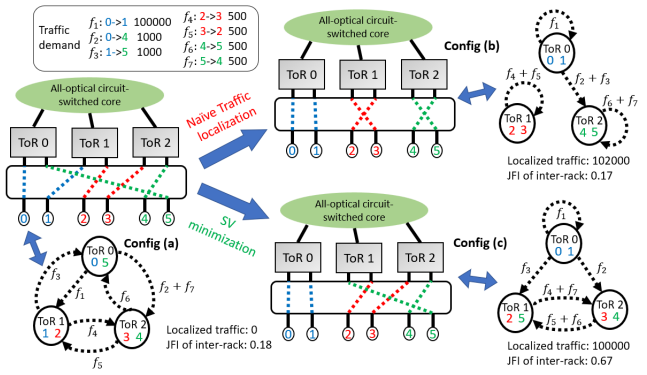

Figure 4: Naïve traffic localization vs. SV minimization: Aggressive localization in config (b) leads to high traffic skewness. Config (c) jointly minimizes traffic skewness and inter-rack traffic volume.

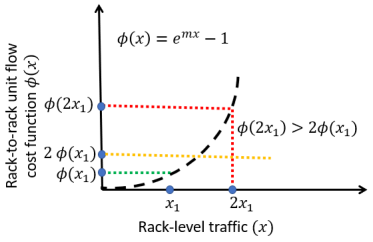

Figure 5: Exponential-based rack-to-rack unit flow cost function can achieve reasonable balance, jointly minimizing skewness and inter-rack traffic volume.

makes the joint optimization problem non-trivial. Consider the example network (Figure 4) having 3 ToRs, with 2 servers each. For simplicity, we assume that there is one edge-OCS, thus any server can move across all the ToR ports. Config (a) shows the initial server-to-ToR mapping and the traffic demand consisting of 7 flows, where f_1 (between servers 0 and 1) is a big flow and the other flows (f_2-f_7) are medium-sized.

For config (a), the traffic is completely inter-rack (localized traffic = 0) alongwith high skewness (Jain's Fairness Index, JFI, of the inter-rack traffic is 0.18). Aggressive traffic localization would localize most of the flows (f_1, f_4, f_5, f_6 and f_7), as shown in config (b). It indeed minimizes the inter-rack traffic volume (localized traffic = 102000). However, it leaves both f_2 and f_3 between ToR 0-ToR 2, leading to high traffic skewness (JFI is 0.17). However, in config (c), only flow f_1 is localized, and other flows are deliberately not localized, rather reorganized between ToRs 1 and 2 in a uniform manner. Moreover, unlike config (b), f_2 and f_3 are now between two different rack pairs (ToR 0-ToR 2, and ToR 0-ToR 1 respectively). This configuration sacrifices localization by a little (localized traffic = 100000) while reducing the inter-rack traffic skewness significantly (JFI is 0.67). Hence, config (c) has the best balance between the objectives, jointly minimizing traffic skewness and inter-rack traffic volume.

The next step is to design a suitable cost function for the SV minimization framework that can obtain the right balance between these intertwined objectives. Let's consider a linear cost function first, i.e. the cost for intra-rack traffic is 0 and the cost of a unit inter-rack traffic flow is constant. This would only achieve traffic localization as it doesn't account for skewness. Considering the example in Figure 4, the linear cost function cannot distinguish the scenarios where f_2 and f_3 are both inter-rack but under the same rack pair (config (b)) vs. under two different rack pairs (config (c)). For both cases, the cost (associated to f_2 and f_3) will be $1000+1000 = 2000$.

Hence, we need a rack-to-rack unit flow cost function that penalizes high variance in inter-rack traffic across different rack pairs. In Figure 4, we want the following condition to be satisfied: $\Phi(2000) > \Phi(1000)+\Phi(1000)$. This can be achieved by a convex non-linear cost function [8]. In this paper, we apply the exponential function $\Phi(x) = e^{mx} - 1$, as shown in Figure 5 (x is the rack-level traffic). If $x = 0$, $\Phi(x) = 0$, i.e., it does not penalize intra-rack traffic. Additionally, it increasingly penalizes larger values of x i.e., $\Phi(2x_1) = e^{2mx_1} - 1 > 2*(e^{mx_1} - 1) = 2*\Phi(x_1)$, thus jointly enforcing inter-rack traffic localization and fairness. Note that, m is the parameter dictating the importance of skewness over inter-rack traffic volume. A large value of m will heavily penalize additional inter-rack traffic, thus minimizing skewness. In our experiments, we found that $m = 2$ achieves a good balance between inter-rack traffic and skewness minimization.

B. SV minimization

Based on the exponential rack-to-rack unit flow cost function (Section V-A), we define the SV minimization problem for dynamic optical-edge reconfiguration for OSSV.

Problem definition: Given the number of ToRs (N), the number of servers per ToR (M), server-level traffic matrix (f), and ToR-level capacity matrix (B), the goal of SV minimization is to find the optimal server-to-ToR mapping (I) that minimizes total cost:

$$cost(f, B, \Phi, I) = \sum_{j=1}^{N}\sum_{i=1}^{N} C_{ij} = \sum_{j=1}^{N}\sum_{i=1}^{N} \Phi_{ij}(F_{ij}, B_{ij})F_{ij} \tag{1}$$

where C is the ToR-level cost matrix and F is the ToR-level traffic matrix i.e., aggregated traffic between the servers assigned to a pair of racks for the given server-to-ToR mapping:

$$F_{ij} = \sum_{s=1}^{MN}\sum_{s'=1}^{MN} I_{si} I_{s'j} f_{ss'} \tag{2}$$

where $I_{si} = 1$, if server s belongs to ToR i in the current server-to-ToR mapping, otherwise $I_{si} = 0$, $f_{ss'}$ is the traffic from server s to s', and Φij is the rack-to-rack unit traffic cost. As discussed in Section V-A, we apply an exponential cost function per unit flow size between rack i to rack j:

$$\Phi_{ij}(F_{ij}, B_{ij}) = e^{m\frac{F_{ij}}{B_{ij}}} - 1 \tag{3}$$

Hardness: SV minimization is equivalent to a variant of the Balanced Graph Partitioning (BGP) problem, which is known to be NP-hard even to approximate by a constant factor [26]. The goal of BGP is to partition a graph $G = (V, E)$, (V is the set of vertices, with $|V| = n$, and E is the set of edges), into k partitions $P_1, P_2, \ldots P_k$ with size n/k such that

the total number of edges connecting different partitions is minimized. We can reduce BGP to SV minimization by (1) representing each vertex in V as a server and each partition P_i as a rack, (2) assigning a unit traffic flow between each pair of servers corresponding to an edge in E, and (3) setting the unit flow size cost function to $\Phi_{ij} = 1, \forall i, j$. Note that SV minimization guarantees that partition sizes are balanced. Moreover, minimizing the cost from Eqn. 1 is equivalent to minimizing the number of edges across partitions.

C. Efficient heuristic design

We implement a hill-climbing based randomized heuristic to solve the SV minimization problem. First, we provide an overview of basic heuristic design and then discuss optimizations to make it more efficient.

Hill-climbing overview: The hill-climbing starts with an initial server-to-ToR mapping. The goal is to find the optimal server-to-ToR mapping in order to minimize the cost function (Eqn 1) for a given server-level traffic matrix (f). Algorithm 1 presents the pseudocode. In each iteration, given a previous server-to-ToR mapping (I^{prev}), it proposes a candidate solution i.e., a new server-to-ToR mapping by randomly swapping two servers (s, s') between two different ToRs (r, r') [lines 1-3] and evaluates the swap by computing the cost difference ($\Delta_{ss'}$ in Eqn 4) [line 4], where I^{new} swaps servers s and s' in I^{prev}. If the cost difference is positive, the swap is accepted, and I^{prev} is updated to I^{new} [lines 5-8], otherwise the swap is rejected. This process is repeated until convergence (if the rejection happens for consecutive p times, it stops).

$$\Delta_{ss'} = cost(f, B, \Phi, I^{prev}) - cost(f, B, \Phi, I^{new}) \quad (4)$$

Algorithm 1 Hill-climbing (f, B, Φ, I^{prev})

1: **for** iterations $1, 2, \ldots T$ **do**
2: Select random ToRs (r, r')
3: Select random servers (s, s') from (r, r') in I^{prev}, resp.
4: $\Delta_{ss'} \leftarrow$ Evaluate-swap s, s' (Eqn 4)
5: **if** $\Delta_{ss'} > 0$ **then**
6: $I^{new} \leftarrow$ swap ToR assignments of s_i, s_j in I^{prev}
7: $I^{prev} \leftarrow I^{new}$
8: **end if**
9: **end for**
10: **return** I^{prev}

Efficient evaluation of swaps: The naïve computation of Eqn 4 takes time $O(M^2N^2)$, where MN is the total number of servers. However, this is redundant because swapping only two servers does not change most of the entries in the F matrix. Hence, we propose a more efficient alternative to compute Eqn 4, which is inspired by the Kernighan-Lin algorithm. Note that, our formulation accounts for non-linear edge cost.

At a given iteration, if server s and s' are swapped between rack r and r', only a subset of flows need to be considered to update the rack-level traffic matrix. More specifically, we can decompose those relevant server-to-server flows into the following categories: (a) outgoing flows from server s except to s' ($f_{sj}|j \neq s'$), (b) outgoing flows from server s' except to s ($f_{s'j}|j \neq s$), (c) incoming flows to server s except from s' ($f_{js}|j \neq s'$), (d) incoming flows to server s' except from s ($f_{js'}|j \neq s$), (e) flow from server s to server s' ($f_{ss'}$), and (f) flow from server s' to server s ($f_{s's}$).

Algorithm 2 provides the pseudocode for efficient computation of Eqn 4. First, we update the F matrix using those relevant server-to-server flows [lines 2-19]. Note that, only the rack-level flows associated with rack r and r' will be affected by the swap. Next, we compute the difference in cost for only those updated elements in F matrix (δ_1-δ_4), and add those to get the resulting cost difference ($\Delta_{ss'}$) for the swap [lines 20-26]. Clearly, the complexity of Algorithm 2 is $O(MN)$, leading to $MN\times$ faster execution than the naïve counterpart.

Algorithm 2 Evaluate-swap $(f, B, \Phi, I^{prev}, F, C, s, s', r, r')$

1: $\Delta_{ss'} = 0$
2: **for** ToR $i \in 1, \ldots N$ **do**
3: **for** server j assigned to ToR i **do**
4: **if** $j \neq s'$ **then**
5: $F_{ri} = F_{ri} - I^{prev}_{ji} * f_{sj}$
6: $F_{r'i} = F_{r'i} + I^{prev}_{ji} * f_{sj}$
7: $F_{ir} = F_{ir} - I^{prev}_{ji} * f_{js}$
8: $F_{ir'} = F_{ir'} + I^{prev}_{ji} * f_{js}$
9: **end if**
10: **if** $j \neq s$ **then**
11: $F_{r'i} = F_{r'i} - I^{prev}_{ji} * f_{s'j}$
12: $F_{ri} = F_{ri} + I^{prev}_{ji} * f_{s'j}$
13: $F_{ir'} = F_{ir'} - I^{prev}_{ji} * f_{js'}$
14: $F_{ir} = F_{ir} + I^{prev}_{ji} * f_{js'}$
15: **end if**
16: **end for**
17: **end for**
18: $F_{rr'} = F_{rr'} - f_{ss'} + f_{s's}$
19: $F_{r'r} = F_{r'r} - f_{s's} + f_{ss'}$
20: **for** ToR $i \in 1, \ldots N$ **do**
21: $\delta_1 = C_{ri} - \Phi_{ri}(F_{ri}, B_{ri})F_{ri}$
22: $\delta_2 = C_{ir} - \Phi_{ir}(F_{ir}, B_{ir})F_{ir}$
23: $\delta_3 = C_{r'i} - \Phi_{r'i}(F_{r'i}, B_{r'i})F_{r'i}$
24: $\delta_4 = C_{ir'} - \Phi_{ir'}(F_{ir'}, B_{ir'})F_{ir'}$
25: $\Delta_{ss'} = \Delta_{ss'} + \delta_1 + \delta_2 + \delta_3 + \delta_4$
26: **end for**
27: **return** $\Delta_{ss'}$

VI. Evaluation

A. Packet-level simulation

We perform simulations extending a packet-level simulator based on htsim [35] which supports TCP congestion control algorithms along with ECMP. We simulate a network of 16 ToR switches, consisting of 32 servers each and 100 Gbps host-transmission speed. We generate the flow-level cache traffic traces [36] (details in Section II-C). In terms of the DCN architectures, we implement a) a round-robin OCS core like Sirius [12] (parameters mentioned in Section II-C), without

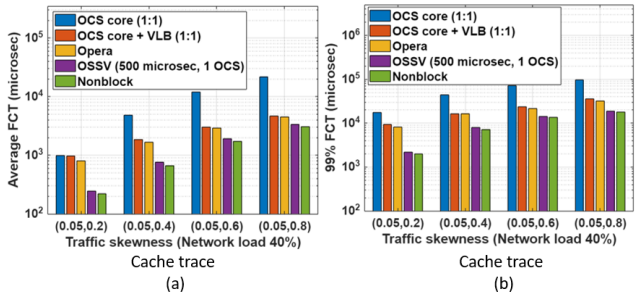

Figure 6: (a) Average and (b) 99% FCT (μsec) at different traffic skewness scenarios [network load 40%, os ratio 1 : 1].

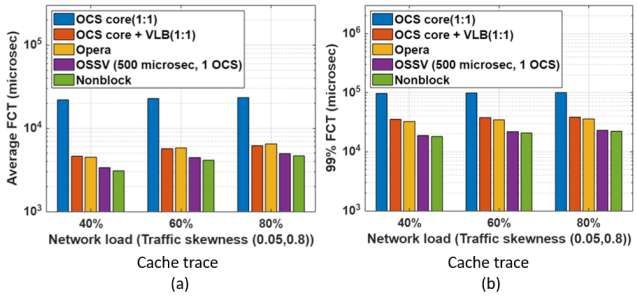

Figure 7: (a) Average and (b) 99% FCT (μsec) at different network loads [skewness $(0.05, 0.8)$, os ratio 1 : 1].

and with valiant load balancing (VLB), that leverages two-hop indirect ToR-level paths under skewed traffic, b) an OCS core going through a series of expander graphs like Opera [29], and c) an ideal packet-switched non-blocking network. For simulating OSSV, we append a 2D-MEMS based OCS edge with a Sirius-like round-robin core (no VLB). We consider three configurations of OSSV data plane: 1) one big edge-OCS with no downtime to get the upper-bound performance, 2) one edge-OCS, and 3) multiple edge-OCS with realistic downtime (10 μsec). We also consider two configurations of OSSV control plane: 1) different reconfiguration epochs, and 2) different edge-topology optimization algorithms (SV minimization, and traffic localization).

B. Impact of skewness

Figures 6(a) and 6(b) show the average and 99% flow completion time (FCT) of the architectures across different traffic skewness, with 1 : 1 os core and 40% network load. At high skewness $(0.05, 0.8)$, the average and 99% FCT slowdown of round-robin OCS core w.r.t. to the non-blocking network is significant (7.08× and 5.36× respectively). VLB can improve the performance to some extent, but the average and 99% FCT slowdowns are still by 50% and 97.9%. Opera primarily optimizes for small flows, leading to 46% and 79% slowdown in average and 99% FCT, respectively. Finally, the upperbound performance of OSSV (one edge-OCS, 10 μsec downtime and 500 μsec epoch interval) is closest to the non-blocking network, the average and 99% FCT slowdown being 9.7% and 5.1% respectively. We observe that 500 μsec is the optimal edge reconfiguration epoch interval for cache traces (Section VI-E).

C. Impact of network load

Next, we vary the network load under high skewness $(0.05, 0.8)$ and core os ratio 1 : 1 (Figures 7(a) and 7(b)). The performance of round-robin core degrades with higher network load, although such degradation happens at a slighly smaller rate compared to that of non-blocking network. At 80% network load, round-robin OCS core has average and 99% FCT slowdown by 5× and 4.5×. For OCS core with VLB and Opera, the 99% FCT can be improved significantly (slowdown being 38% and 61% respectively). OSSV can further improve the performance, making 99% FCT within 4% of the ideal non-blocking network.

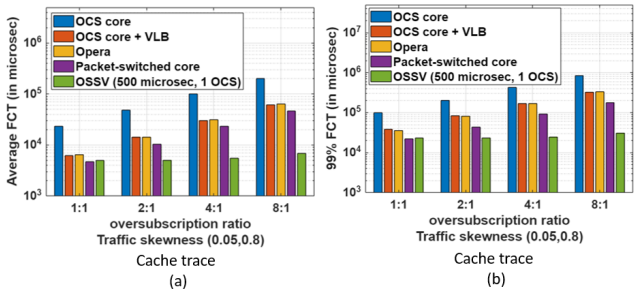

Figure 8: (a) Average and (b) 99% FCT (μsec) at different core os ratios [skewness $(0.05, 0.8)$, network load (80%)].

D. Impact of network oversubscription

Figures 8(a) and 8(b) show that, the performance of baseline architectures can be severely affected by higher core oversubscription in presence of high traffic skewness and heavy inter-rack traffic volume. At os ratio 8 : 1, round-robin OCS core has 43.6× worse average FCT compared to a non-blocking network. At such high os ratio, VLB and Opera both improve upon OCS-based core, but still have average FCT slowdown of 13.12× and 13.8×, respectively. A traditional packet-switched network can outperform all of these baseline architectures at high os ratio, due to more path diversity and no OCS-based downtime. However, OSSV can improve the performance significantly compared to a packet-switched network at high os ratio, as SV minimization can obtain the right balance between the two objectives depending on available bandwidth at the core. For example, at 1 : 1 os ratio, SV minimization localizes 14% of the overall traffic having inter-rack JFI of 0.98. However, at 8 : 1 os ratio, SV minimization localizes 56% of overall traffic, while maintaining inter-rack JFI of 0.97, which effectively reduces the core os ratio from 8 : 1 to 1.4 : 1.

E. Impact of reconfiguration epoch interval in OSSV

Figures 9(a) and 9(b) show the average and 99% FCT of OSSV normalized w.r.t. a non-blocking network at skewness (0.05, 0.8), varying edge reconfiguration epoch interval with and without considering the edge-OCS downtime. Ideally, if the edge OCS would have a downtime-related penalty, the performance of OSSV would improve with a smaller reconfiguration epoch interval. However, in presence of OCS downtime, we observe a sweet spot in terms of reconfiguration epoch interval, achieving the best trade-off between observation frequency vs. reconfiguration penalty. For cache traffic, we

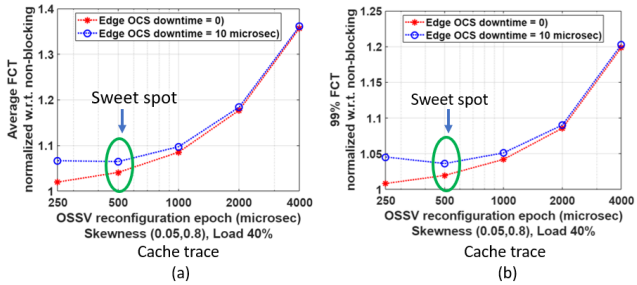

Figure 9: (a) Average and (b) 99% FCT (μsec) of OSSV normalized w.r.t. nonblocking at different edge reconfiguration epoch intervals without and with physical downtime (10 μsec).

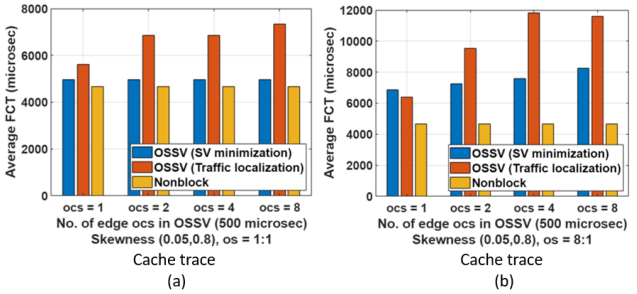

Figure 10: Average FCT of OSSV with different no. of edge-OCS at os ratios (a) $1:1$, and (b) $8:1$, with SV minimization vs. traffic localization.

observe the optimal epoch interval is 500 μsec, which remains the same across traffic skewness and network loads. Similar data-driven analysis can be performed to obtain the suitable edge reconfiguration frequency for other workloads.

F. Impact of multiple edge-OCS with different cost functions

Figures 10(a) and 10(b) demonstrate the impact of multiple small port-count edge-OCS in OSSV, at os ratio $1:1$ and $8:1$, respectively, under high skewness $(0.05, 0.8)$ and heavy inter-rack traffic volume (98%). OSSV with SV minimization has very little penalty across multiple edge-OCS at $1:1$ os. Even with 8 edge-OCS, the 99% FCT degrades by only 2.19% compared to 1 edge-OCS. To understand the reason, we observe the % of localized traffic and JFI of inter-rack traffic across multiple edge-OCSes. At $1:1$ os ratio, the % of localized traffic varies from 14% to 9%, while JFI index of inter-rack traffic varies from 0.98 to 0.93. However, the performance of OSSV with naïve traffic localization suffers under multiple edge-OCS even with $1:1$ os (29.6% performance penalty from 1 edge-OCS to 8 edge-OCS). Clearly, the aggressive localization reduces inter-rack traffic volume significantly (% of localized traffic is 64%), but the remaining inter-rack traffic becomes highly skewed (JFI ranges from $0.21 - 0.26$). A similar trend is observed under $8:1$ os, where the performance degradation for naïve traffic localization is more than 81%, but for SV minimization it is within 20.5%.

G. Benchmarking the SV minimization

Figure 11(a) shows that, under skewed traffic $(0.05, 0.8)$ and high network load (80%), the hill-climbing for the SV minimization has fast-convergence (within a few thousands of iterations). We also measure the speedup of hill-climbing with

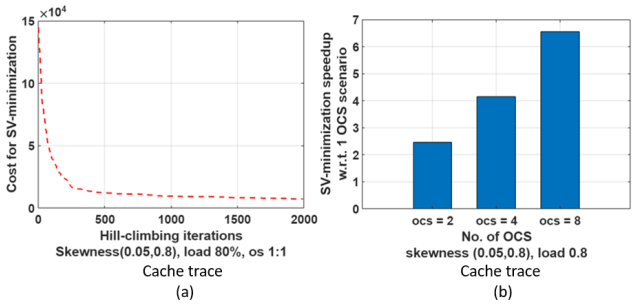

Figure 11: (a) Hill-climbing convergence for SV minimization, (b) Trade-off between edge-OCS port count vs. SV minimization speedup.

Components	Power (Watt)	Cost (USD)
Ethernet port (100G) [4]	14.1	293
Ethernet port (400G) [5]	39.4	859
OCS core port (grating) [12]	0	$5\% - 100\%$ of Ethernet port cost (100G)
OCS edge port (MEMS) [3]	0.14	100
Fixed optical transceiver [2]	3.5	399
Tunable optical transceiver [12]	$1 - 10\times$ of fixed transceiver power	$5\times$ of fixed transceiver cost
Intra-rack fiber [7]	0	6.9
Inter-rack fiber [6]	0	4.9

Table I: Power/cost data of the components for OSSV, OCS core (like Sirius), and packet-switched core (100 Gbps).

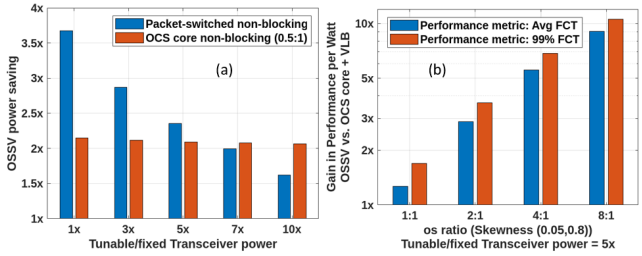

Figure 12: (a) OSSV (os $1:1$) power saving w.r.t. packet-switched and OCS core-based non-blocking networks, varying tunable/fixed transceiver power ratio [12], (b) Gain in performance per watt [skewness $(0.05, 0.8)$, network load 80%]: OSSV vs. OCS-based core with VLB at different os ratios.

optimized vs. naïve versions. On average, the speed up is more than $60\times$. Figure 11(b) shows the trade-off between edge-OCS port count vs. SV minimization runtime. Compared to 1 edge-OCS case, the speedups are $2.4\times$, $4.1\times$ and $6.6\times$ for 2, 4, and 8 edge-OCS cases, respectively. Hence, OSSV control plane can scale with multiple edge-OCS leading to reduced control overhead, while maintaining high performance.

H. Power and cost analysis

We perform a detailed power consumption and capital cost analysis (independent of network size) of OSSV, and compare it with a) packet-switched non-blocking core, and b) Sirius-like OCS-based network core under different os ratios. Table I summarizes the relevant components along with the power and cost values at 100 Gbps base data rate. Note that, to model the non-blocking OCS-core with VLB (i.e., core os $0.5:1$), we consider the next available 400 Gbps Ethernet switch, with each uplink port having two break-out cables and two 100 Gbps tunable optical transceivers, thus each logical uplink operates at 200 Gbps.

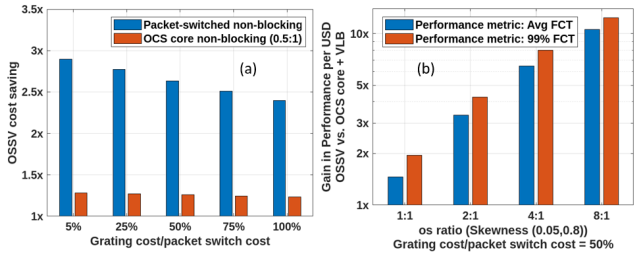

Figure 13: (a) OSSV (os $1:1$) cost saving w.r.t. packet-switched and OCS core-based non-blocking networks, varying the grating/packet-switch cost ratio [12], (b) Gain in performance per USD [skewness $(0.05, 0.8)$, network load 80%]: OSSV vs. OCS-based core with VLB at different os ratios.

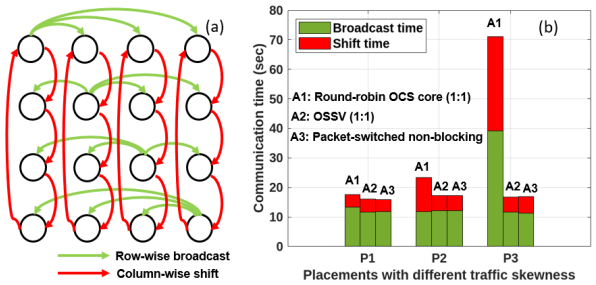

Figure 14: (a) DMM traffic pattern, (b) Average broadcast and shift time of the architectures (testbed prototype) under three placements resulting in different traffic skewness.

We vary two different key dimensions: tunable transceiver power and grating cost [12]. OSSV (os $1:1$) can be up to $3.6\times$ and $2.1\times$ more power-efficient (Figure 12(a)) w.r.t. packet-switched and OCS-based non-blocking cores, respectively, across different tunable laser powers. OSSV has higher performance per watt ($1.26\times - 9\times$) w.r.t. OCS-based core with VLB at different os ratios (Figure 12(b)). In Figure 13, we consider the tunable laser cost as $5\times$ of fixed lasers [12], 2D-MEMS based OCS cost is upper-bounded by 3D-MEMS based OCS [33], and vary the grating cost. OSSV (os $1:1$) can be up to $2.89\times$ and $1.28\times$ more cost-efficient (Figure 13(a)) compared to packet-switched and OCS-based non-blocking cores, respectively, across different grating costs. Also, OSSV has higher performance per USD ($1.46\times - 10.6\times$) w.r.t. OCS-based core with VLB at different os ratios (Figure 13(b)).

I. Testbed evaluation

We implement the prototypes of round-robin OCS core (os $1:1$), OSSV (os $1:1$), and packet-switched non-blocking network with 16 servers and 4 racks. Next, we run OpenMPI based DMM algorithm [19] with 16 processes (each process under one server). In each iteration, the algorithm goes through a "broadcast-shift-multiply" cycle (Figure 14(a)), where a process performs row-wise broadcast and column-wise shift of submatrices, followed by a local multiplication.

In our experiment, we consider three potential process placements $P1$, $P2$, and $P3$, leading to different extent of traffic skewness. In $P1$, broadcast is rack-to-rack shuffle and shift is intra-rack, leading to uniform traffic. In $P2$, broadcast is intra-rack but the shift is inter-rack leading to moderate traffic skewness. Finally, in $P3$, both broadcast and shift have high rack-level imbalance, leading to a high degree of overall skewness. As shown in Figure 14(b), under $P1$, all three architectures have similar performance as the traffic is uniform. Under $P2$, OSSV improves the average shift time of the round-robin core by $2.28\times$, which is close to ideal (non-blocking network). Finally, under $P3$, performance of the round-robin OCS core significantly suffers from high skewness during both broadcast and shift. OSSV, equipped with dynamic edge topology reconfiguration, can improve the average broadcast and shift time by $3.36\times$ and $6.27\times$ respectively.

VII. Related Work

Existing round-robin OCS cores (RotorNet [30], Sirius [12] etc.) are not suitable for diverse DCN workloads with high skewness and large inter-rack traffic volume. Valiant load balancing [39] can partially solve the skewness issue using a two-hop indirect path, but the worst-case throughput can be 50%. Moreover, VLB cannot reduce the inter-rack traffic volume. Opera [29] extends RotorNet and improves the performance by allowing instant transmission of latency-sensitive flows through multi-hop expanders. However, Opera has higher average path length which could be severely harmful, if the traffic is heavily skewed and inter-rack under high load and oversubscription scenarios. OSSV, with a combination of traffic-agnostic round-robin optical core and traffic-aware reconfigurable optical edge, proposes a new paradigm for all-optical DCN architectures. Several rack-level reconfigurable networks [18], [24], [27], [40], [43] provision bandwidth on demand to optimize for dynamic workloads. But they add extra bandwidth at the ToR-level through ad-hoc links. OSSV provides runtime flexibility at the edge and makes SV minimization without adding any extra bandwidth. [16], [17] made a case that traffic skewness could be harmful to OCS-based network cores, which was a partial understanding of the problem. In contrast, this paper identifies the interplay between traffic skewness and inter-rack traffic volume, develops a concrete problem formulation, provides a novel optimization framework, and thoroughly evaluates the proposal.

VIII. Conclusion

OCS is a promising building block for the energy-efficient future generation DCN. However, proposed OCS-based architectures can deliver their promised benefits only if they can efficiently handle diverse DCN workloads having high skewness and inter-rack traffic volume. In this paper, we propose OSSV, a combination of traffic agnostic optical core alongwith a reconfigurable optical edge that jointly minimizes traffic skewness and inter-rack traffic volume. OSSV can significantly reduce the performance gap between OCS-based and ideal packet-switched nonblocking DCN arhitectures, thus making OCS-based architectures widely deployable.

Acknowledgement

We thank the anonymous reviewers for their insightful feedback. This work is partially supported by the NSF under CNS-2214272 and CNS-1815525.

REFERENCES

[1] HDFS architecture. https://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html, 2020.

[2] 100G QSFP28 1310nm 10km transceiver. https://www.fs.com/products/102531.html, 2023.

[3] 320x320 3D MEMS optical circuit switch. https://www.calient.net/, 2023.

[4] 32*100Gbps Ethernet Switch. https://www.fs.com/products/110480.html, 2023.

[5] 32*400Gbps Ethernet Switch. https://www.fs.com/products/149853.html, 2023.

[6] Duplex single mode optical fiber cable (10m). https://www.fs.com/products/40203.html, 2023.

[7] Duplex single mode optical fiber cable (3m). https://www.fs.com/products/40193.html, 2023.

[8] Basic propertie of convex functions. https://wiki.math.ntnu.no/_media/tma4180/2016v/note2.pdf, 2026.

[9] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 265–283, 2016.

[10] M. Al-Fares, S. Radhakrishnan, B. Raghavan, N. Huang, and A. Vahdat. Hedera: dynamic flow scheduling for data center networks. In *NSDI*, volume 10, pages 89–92, 2010.

[11] M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan. Data center tcp (dctcp). In *Proceedings of the ACM SIGCOMM 2010 Conference*, pages 63–74, 2010.

[12] H. Ballani, P. Costa, R. Behrendt, D. Cletheroe, I. Haller, K. Jozwik, F. Karinou, S. Lange, K. Shi, B. Thomsen, et al. 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*, pages 782–797, 2020.

[13] T. Benson, A. Akella, and D. A. Maltz. Network traffic characteristics of data centers in the wild. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, pages 267–280, 2010.

[14] T. Benson, A. Anand, A. Akella, and M. Zhang. Understanding data center traffic characteristics. In *Proceedings of the 1st ACM workshop on Research on enterprise networking*, pages 65–72. ACM, 2009.

[15] P. Bodík et al. Surviving failures in bandwidth-constrained datacenters. In *SIGCOMM*. ACM, 2012.

[16] S. Das, A. Silva, and T. S. E. Ng. Poster: Near non-blocking performance with all-optical circuit-switched core. In *Proceedings of the ACM SIGCOMM 2023 Conference*, pages 1117–1119, 2023.

[17] S. Das, W. Wang, and T. S. E. Ng. Towards all-optical circuit-switched datacenter network cores: The case for mitigating traffic skewness at the edge. In *Proceedings of the ACM SIGCOMM 2021 Workshop on Optical Systems*, pages 1–5, 2021.

[18] N. Farrington, G. Porter, S. Radhakrishnan, H. H. Bazzaz, V. Subramanya, Y. Fainman, G. Papen, and A. Vahdat. Helios: a hybrid electrical/optical switch architecture for modular data centers. *ACM SIGCOMM Computer Communication Review*, 40(4):339–350, 2010.

[19] G. Fox, S. Otto, and A. Hey. Matrix algorithms on a hypercube i: Matrix multiplication. *Parallel Computing*, 4(1):17 – 31, 1987.

[20] P. X. Gao, A. Narayan, S. Karandikar, J. Carreira, S. Han, R. Agarwal, S. Ratnasamy, and S. Shenker. Network requirements for resource disaggregation. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 249–264, 2016.

[21] M. Ghobadi, R. Mahajan, A. Phanishayee, N. Devanur, J. Kulkarni, G. Ranade, P.-A. Blanche, H. Rastegarfar, M. Glick, and D. Kilper. Projector: Agile reconfigurable data center interconnect. In *Proceedings of the 2016 ACM SIGCOMM Conference*, pages 216–229, 2016.

[22] R. Grandl, G. Ananthanarayanan, S. Kandula, S. Rao, and A. Akella. Multi-resource packing for cluster schedulers. *ACM SIGCOMM Computer Communication Review*, 44(4):455–466, 2014.

[23] Q. Huang, H. Gudmundsdottir, Y. Vigfusson, D. A. Freedman, K. Birman, and R. van Renesse. Characterizing load imbalance in real-world networked caches. In *Proceedings of the 13th ACM Workshop on Hot Topics in Networks*, pages 1–7, 2014.

[24] S. Kandula, J. Padhye, and P. Bahl. Flyways to de-congest data center networks. 2009.

[25] S. Kassing, A. Valadarsky, G. Shahaf, M. Schapira, and A. Singla. Beyond fat-trees without antennae, mirrors, and disco-balls. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, pages 281–294, 2017.

[26] R. Krauthgamer, J. Naor, and R. Schwartz. Partitioning graphs into balanced components. In *Proceedings of the twentieth annual ACM-SIAM symposium on Discrete algorithms*, pages 942–949. SIAM, 2009.

[27] H. Liu, F. Lu, A. Forencich, R. Kapoor, M. Tewari, G. M. Voelker, G. Papen, A. C. Snoeren, and G. Porter. Circuit switching under the radar with reactor. In *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*, pages 1–15, Seattle, WA, 2014. USENIX Association.

[28] Z. Liu, Z. Bai, Z. Liu, X. Li, C. Kim, V. Braverman, X. Jin, and I. Stoica. Distcache: Provable load balancing for large-scale storage systems with distributed caching. In *17th USENIX Conference on File and Storage Technologies (FAST 19)*, pages 143–157, 2019.

[29] W. M. Mellette and R. Das. Expanding across time to deliver bandwidth efficiency and low latency. In *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2020.

[30] W. M. Mellette, R. McGuinness, A. Roy, A. Forencich, G. Papen, A. C. Snoeren, and G. Porter. Rotornet: A scalable, low-complexity, optical datacenter network. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, pages 267–280, 2017.

[31] R. Miao, H. Zeng, C. Kim, J. Lee, and M. Yu. Silkroad: Making stateful layer-4 load balancing fast and cheap using switching asics. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, pages 15–28. ACM, 2017.

[32] P. Pannuto, G. Papen, G. Porter, B. Raghavan, A. Schulman, and A. C. Snoeren. C3 lab: Integrated innovation for de-carbonizing datacenters. 2021.

[33] C. Pollock, F. Pardo, M. Imboden, and D. Bishop. Open loop control theory algorithms for high-speed 3d mems optical switches. *Optics Express*, 28(2):2010–2019, 2020.

[34] G. Porter, R. Strong, N. Farrington, A. Forencich, P. Chen-Sun, T. Rosing, Y. Fainman, G. Papen, and A. Vahdat. Integrating microsecond circuit switching into the data center. *ACM SIGCOMM Computer Communication Review*, 43(4):447–458, 2013.

[35] C. Raiciu, C. Paasch, S. Barre, A. Ford, M. Honda, F. Duchene, O. Bonaventure, and M. Handley. How hard can it be? designing and implementing a deployable multipath tcp. In *9th USENIX symposium on networked systems design and implementation (NSDI 12)*, pages 399–412, 2012.

[36] A. Roy, H. Zeng, J. Bagga, G. Porter, and A. C. Snoeren. Inside the social network's (datacenter) network. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, pages 123–137, 2015.

[37] V. Shrivastav, A. Valadarsky, H. Ballani, P. Costa, K. S. Lee, H. Wang, R. Agarwal, and H. Weatherspoon. Shoal: A network architecture for disaggregated racks. In *16th USENIX Symposium on Networked Systems Design and Implementation (NSDI 19)*, pages 255–270, 2019.

[38] A. Singh, J. Ong, A. Agarwal, G. Anderson, A. Armistead, R. Bannon, S. Boving, G. Desai, B. Felderman, P. Germano, et al. Jupiter rising: A decade of clos topologies and centralized control in google's datacenter network. *ACM SIGCOMM computer communication review*, 45(4):183–197, 2015.

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

[40] G. Wang, D. G. Andersen, M. Kaminsky, K. Papagiannaki, T. S. E. Ng, M. Kozuch, and M. Ryan. c-through: Part-time optics in data centers. In *ACM SIGCOMM Computer Communication Review*, volume 40, pages 327–338. ACM, 2010.

[41] W. Wang, D. Wu, S. Das, A. Rahbar, A. Chen, and T. S. E. Ng. RDC: Energy-efficient data center network congestion relief with topological reconfigurability at the edge. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*, pages 1267–1288, 2022.

[42] D. Wu, Y. Xia, X. S. Sun, X. S. Huang, S. Dzinamarira, and T. S. E. Ng. Masking failures from application performance in data center networks with shareable backup. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*, pages 176–190, 2018.

[43] X. Zhou, Z. Zhang, Y. Zhu, Y. Li, S. Kumar, A. Vahdat, B. Y. Zhao, and H. Zheng. Mirror mirror on the ceiling: Flexible wireless links for data centers. *ACM SIGCOMM CCR*, 42(4):443–454, 2012.