Semi-Oblivious Reconfigurable Datacenter Networks

Nitika Saran
Cornell University
Ithaca, NY, USA

Daniel Amir
Cornell University
Ithaca, NY, USA

Tegan Wilson*
Northeastern University
Boston, MA, USA

Robert Kleinberg
Cornell University
Ithaca, NY, USA

Vishal Shrivastav
Purdue University
West Lafayette, IN, USA

Hakim Weatherspoon
Cornell University
Ithaca, NY, USA

Abstract

Reconfigurable datacenter networks use fast optical circuit switches to provide high bandwidths at low cost, therefore emerging as a compelling alternative to packet switching. These switches offer micro- and nano-second reconfiguration, and reacting to demand at this time scale is infeasible. Proposed designs have therefore largely been oblivious, supporting arbitrary traffic patterns. However, this imposes a fundamental latency-throughput tradeoff that significantly limits the benefits of these switches.

In this paper, we illustrate the feasibility of semi-oblivious reconfigurable datacenter networks that periodically adapt to large-scale structural patterns in traffic. We argue that such patterns are predictable in modern datacenters, that optimizing for them can provide latency-throughput scaling superior to oblivious designs, and that existing fast circuit-switched technologies support coarse-grained flexibility to adapt to these patterns.

CCS Concepts

• **Networks** → **Network architectures**; **Data center networks**.

Keywords

Optical Switches, Datacenter Networks

ACM Reference Format:

Nitika Saran, Daniel Amir, Tegan Wilson, Robert Kleinberg, Vishal Shrivastav, and Hakim Weatherspoon. 2024. Semi-Oblivious Reconfigurable Datacenter Networks. In *The 23rd ACM Workshop on Hot Topics in Networks (HOTNETS '24), November 18–19, 2024, Irvine, CA, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3696348.3696860

*Work done in part while at Cornell University.

HOTNETS '24, November 18–19, 2024, Irvine, CA, USA

ACM ISBN 979-8-4007-1272-2/24/11
https://doi.org/10.1145/3696348.3696860

1 Introduction

Over the last two decades, datacenters have seen exponential growth in their bandwidth requirements. Providing petabit connectivity between 10,000s of servers, as switching chip manufacturers contend with the end of Moore's law, has made it increasingly challenging to contain the costs and complexity of datacenter networks (DCNs). Operators and academics alike have therefore looked beyond the paradigm of Clos-based hierarchical packet-switched networks.

Optical circuit switches (OCSes) have remained a popular alternative for two reasons: they fundamentally lower cost and power consumption by reducing electronic components, and can reconfigure the topology to effectively use bandwidth. Fast circuit switches with microsecond and nanosecond reconfiguration times, can scale these benefits by replacing most of the electrical network. [5, 18, 20]

To avoid predicting dynamic datacenter workloads at nanosecond time scales, fast circuit-switched proposals have been primarily demand-oblivious, supporting arbitrary traffic patterns via uniform connectivity. Remaining completely oblivious, however, imposes a fundamental latency-throughput tradeoff: we must either split each node's bandwidth uniformly across the network, inflating latency, or route flows via multiple indirect hops, inflating required bandwidth [4].

In this paper, we argue that reconfigurable datacenters need not succumb to this obliviousness barrier. While it is hard to predict *micro*-scale patterns, such as individual flows or precise bandwidth requirements between hosts or racks, DCN traffic exhibits predictable *macro*-scale structural patterns, such as spatial locality and flow-size distributions. Such patterns depend on the nature and distribution of target applications, which are relatively stable over time, and can be inferred by application-level control plane entities such as schedulers or job placement systems [12, 23, 29]. Request and utilization patterns at the application level may then offer several dimensions of predictability for the network.

To take advantage of such structural predictability, we propose *semi-oblivious* reconfigurable networks, which periodically update the topology to match macro-patterns in the

datacenter. We show that accounting for this coarse-grained information allows efficient use of network bandwidth, to simultaneously achieve high throughput and low latency.

We describe two commonly found stable patterns in the datacenter: a known degree of spatial locality, and aggregated traffic matrices between clusters of racks/machines. Using these patterns, we construct a semi-oblivious reconfigurable network, that achieves latency-throughput scaling superior to existing systems - maintaining throughput similar to fully connected oblivious designs [5, 20, 27], while lowering latency by orders of magnitude at datacenter scale. We also show that existing fast circuit switching technologies offer a path to periodically update the circuit schedule, and can support coarse-grained flexibility to match structural patterns. Finally, these insights lead to significant potential for future directions in both systems and algorithm design. We therefore end with a detailed discussion on implications and open questions.

2 Reconfigurable Networks

Optical Circuit Switches (OCSes) establish high-bandwidth photonic circuits between input and output ports. By operating entirely in the optical domain, OCSes eliminate the need for CMOS chips and optical transceivers, reducing power consumption by an order of magnitude compared to packet switches at equivalent bitrates [27]. Reconfigurable networks leverage these switches to construct high-bandwidth, dynamic network topologies at low costs.

Cloud operators report that with traffic doubling every year, a datacenter now must support bisection bandwidth equivalent to the entire Internet [1]. As these demands continue to grow, and Moore's nears its end, packet switches are already reaching their performance limits. Optical circuit switching, offers a rare opportunity for fundamental reductions by lowering per-bit cost and power consumption, while also efficiently adapting bandwidth to dynamic workloads. Industrial deployments of reconfigurable networks report CapEx and OpEx reductions of about 30%, along with simplified evolution of the core network [17, 22].

Fast Circuit Switches. While OCSes with millisecond reconfiguration times are beneficial at small scales — handling bulk transfers and traffic hotspots [13, 32], or operating at the spine layer—large [22], datacenter networks (DCNs) with thousands of racks or servers must also accommodate latency-sensitive bursts of traffic. Fast circuit switching, with microsecond (μs) or nanosecond (ns) reconfiguration, enables high-fanout at each node by time-sharing ports across a schedule of circuits. This effectively uses node bandwidth to communicate with many other nodes simultaneously, serving both latency-sensitive and bulk traffic. These fast optical circuit switches can potentially reduce DCN costs by up to

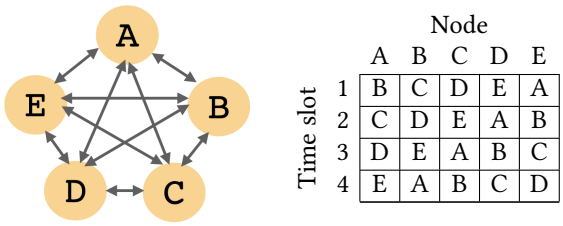

Figure 1: An oblivious reconfigurable network for 5 nodes, with a round-robin schedule of connections.

70% [5], spurring several research and industrial efforts to manufacture and deploy them at scale [1, 19].

Precise Demand Fitting is Infeasible. Early designs using μs-scale circuit switches proposed μs control planes to predict and schedule flows in the network [10, 21]. But not only is such a tightly synchronized control plane infeasible to implement at scale, datacenter flows are not predictable. Datacenters host a variety of applications and services, and communication is often bursty, making fine-grained bandwidth requirements at the server or rack level unpredictable. Further, matching the topology to a given traffic matrix is sometimes impossible. Choosing arbitrary topologies between N nodes requires $N!$ physical configurations in the OCS layer. For many OCS designs, physical constraints prevent them from simultaneously achieving many configurations, fast switching, and low loss [19, 20]. Arbitrary flexibility between 1000s of racks is also not usually required. Operators prefer intuitive topologies for robustness to demand changes, and to quickly reason about failures and fix outages [17, 22].

Oblivious Reconfigurable Networks (ORNs). ORNs avoid a demand-aware control loop by targeting uniform connectivity, and supporting arbitrary traffic with worst-case throughput guarantees. Nodes and switches synchronously cycle through a predetermined schedule of circuits to create a fixed logical topology. A common strategy is to use a round-robin schedule at each node, implementing full connectivity as in Figure 1 [5, 20, 27]. If traffic was uniformly all-to-all, single-hop paths best use bandwidth and minimize latency. But to support arbitrary demands, these designs use Valiant Load balancing (VLB), spreading traffic across 1-hop indirect paths for a worst-case throughput of 50% [31].

Fundamental Latency-Throughput Tradeoff. For moderately sized networks up to a few hundred nodes, round robin schedules effectively split bandwidth across every source-destination pair. A high fanout at nodes leverages statistical multiplexing for latency-sensitive traffic, implementing a cost-efficient clique. But as the circuit schedule scales linearly with network size, the time to cycle through this becomes prohibitively high, even with nanosecond-scale switching. For example, for 10,000 nodes, a round robin schedule with $50ns$ time slots can take $500\mu s$ to cycle through.

A packet may then be subjected to this delay, even before accounting for queuing and propagation!

To shorten the circuit schedule and cycle time per node, Opera [18] uses expander graphs between ToR switches, routing traffic over multiple indirect hops. But increasing indirect hops proportionally increases total traffic volume, sacrificing network throughput. This results in a high *bandwidth tax* or overprovisioning cost, limiting the low-latency advantage to a small portion of traffic (<10%). Prior works [4, 35] show a Pareto optimal latency-throughput tradeoff by parameterizing the round-robin schedule across h dimensions: an h-dimensional optimal ORN uniformly routes traffic on $2h$ hops for worst-case latency $O(h\sqrt[h]{N})$, and worst-case throughput $1/2h$. However, this introduces an inherent scaling barrier for ORNs. For instance, a 2D optimal ORN reduces latency exponentially with 4-hop routing, compared to a flat round robin (1D optimal ORN), but throughput drops to 25% [3]. This imposes significant constraints on the cost and power savings of optical switching, even with *ns* circuit switches.

Since oblivious reconfigurable networks commit to a fixed circuit schedule and make no assumptions about traffic, they must provide full uniform connectivity at all time — leading to the tradeoff described above. However, this approach is overly maximalist. While precise traffic matrices are not predictable, datacenter traffic is structured with macro-scale patterns, that can be exploited in the circuit schedule. Additionally, while adaptive control loops are too slow to respond to micro-scale patterns such as individual flows or microbursts, the macro-patterns in datacenters are often stable over time [22, 23].

We therefore propose semi-oblivious reconfigurable networks (SORNs), which can periodically adjust their topology to match structural patterns in traffic demand. By optimizing for structural macro-patterns, we argue that SORNs can effectively use bandwidth to match throughput of fully-connected ORNs, while lowering latency by orders of magnitude. Evaluating the feasibility of such networks requires understanding macro-patterns in datacenter workloads, and corresponding reconfiguration mechanisms for circuit schedules. We address both of these in turn.

3 Large-Scale Traffic Patterns

Datacenter network traffic varies widely with operator use case, and requirements change over time. For static network topologies, supporting arbitrary workloads is necessary to avoid restricting applications. Existing DCNs therefore usually provide uniform connectivity, even as cloud operators widely acknowledge that traffic is far from all-to-all, which in turn leads to huge bandwidth overprovisioning [6, 23]. Reconfigurable networks can, in theory, match topology to demand, but traffic patterns targeted so far are too short-lived [32, 36]. By the time a network control plane can detect changes, and reconfigure the topology, the demand may have already changed. For example, hotspots and bulk transfers are susceptible to bursts, where past behavior often fails to predict the future. This limits the benefits of reconfiguration, subjecting fast circuit switches to the same overprovisioning constraints as static designs.

At the same time, common applications in the datacenter can be characterized by their network usage. For instance, user-facing web services, data mining and machine learning, and caches all exhibit specific latency and bandwidth requirements. The distribution of datacenter workload across these services varies, but is often stable over hours or days, and can be inferred at the control plane. These applications also have distinctive hardware requirements, which can lead to a predictable spatial distribution for them. Machines or racks in a datacenter are usually arranged into a spatial hierarchy of pods, clusters, or blocks to facilitate management [11, 23, 28]. Scheduling and job placement algorithms use this hierarchy to map requests to resources, and similar applications may be co-located due to coinciding resource requirements. Authors in [23], for example, describe that clusters of machines serve distinct roles. Even for public clouds hosting virtual machines or external services, network requirements can usually be put in the context of this spatial hierarchy. For instance, requests specify preferences for co-location within different hierarchical levels, and operators report that the distribution of these requests is predictable [12].

We interpret this predictability of application-level workload as macro-scale structural suggestions for the network. Specifically, we envision aggregating nodes (end-hosts / ToRs) into "cliques" of variable size. Depending on application workload, these cliques may exhibit high density among themselves but more importantly represent stable aggregate demand patterns between groups of nodes. We describe both of these target patterns below, in the context of production traffic reports.

Spatial Locality. Production datacenter measurements capture varying degrees of traffic locality within every level of the spatial hierarchy. This locality may vary over time and across the network but is predictable based on application workload [7, 8, 23, 28]. We therefore assume that co-located nodes can be grouped into cliques, with a known degree of locality within them. Importantly, even in the absence of traffic locality, i.e. uniform all-to-all traffic, the network can still be optimized accordingly.

Aggregated Traffic Matrices. Optimizing for traffic matrices between individual racks/servers is hard due to the number of source-destination pairs, and bursts of transfers between each pair. However, traffic patterns between groups of 100s of machines show much more stability and predictability. In some cases, this comes from pre-assigned roles

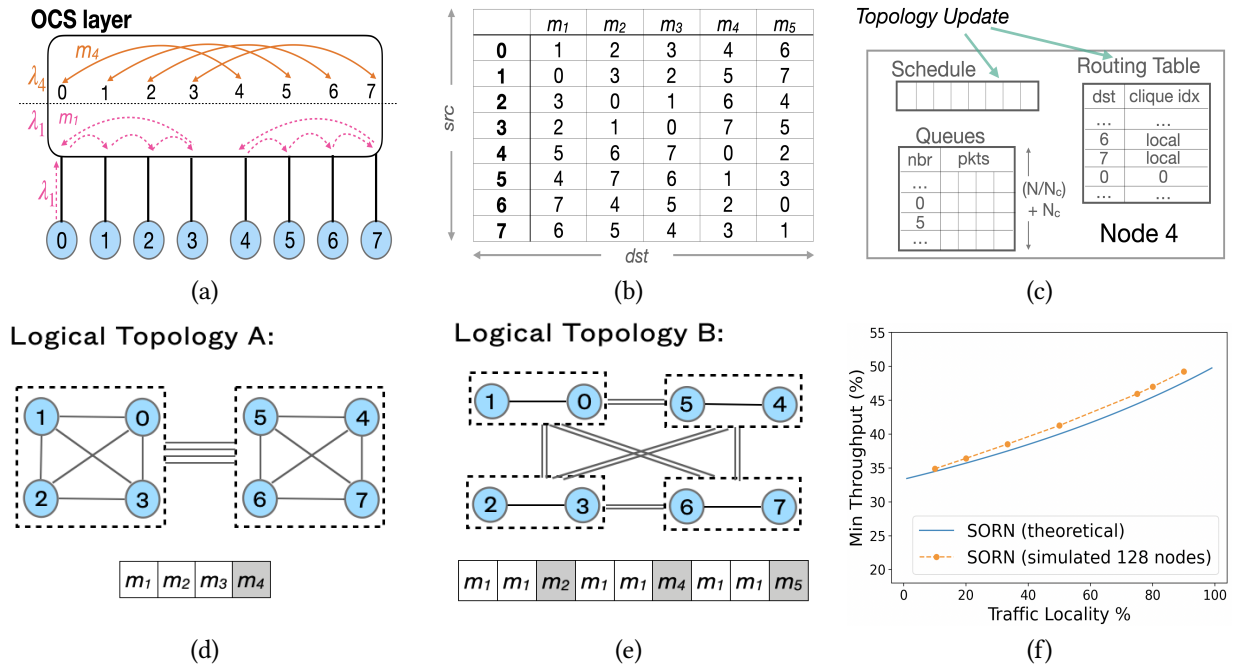

Figure 2: (a) An example optical circuit switched setup enabling several matchings between 8 nodes via wavelength-based routing; (b) enlists matchings; (c) a schematic for node state showing how circuit schedule can be updated; (d) and (e) show different logical topologies achievable. (f) Worst-case throughput for our semi-oblivious design with varying traffic locality ratios.

to certain machines [23]. Other production DCNs report stable gravity traffic patterns between clusters of machines [22]. In this case, grouping nodes leverages statistical multiplexing to smooth bursts between individual nodes, such that the aggregate bandwidth requirements are relatively stable. Google's Jupiter [22] for example, performs well by infrequently adapting to such aggregated patterns. We therefore also assume by grouping nodes into cliques, bandwidth requirements across these cliques are roughly predictable.

4 Semi-Oblivious Reconfigurable Network

Based on the patterns described in the previous section, we propose a semi-oblivious reconfigurable network (SORN). This implements a hierarchical but reconfigurable topology, with cliques of nodes. Within these cliques, we implement uniform bandwidth density to support arbitrary traffic but set stable bandwidth requirements across these cliques. As workload needs vary, we may update the size of these cliques, the oversubscription of bandwidth between them, and the aggregate topology across cliques, but these parameters are assumed stable over long periods. In this section, we show how fast circuit switches can implement such macro-patterns, describe one possible routing scheme, and show cost and performance benefits over oblivious designs.

We consider a wavelength-selective OCS setup similar to Sirius [5], an abstract representation of which is shown in Figure 2 (a) and (b) for an 8-node topology. The nodes could be either ToR switches or end-hosts of a DCN. Nodes connect to an optical circuit switched layer, that provides a set of *matchings* between input and output ports. This is a similar abstraction to other reconfigurable DCN designs [18, 20]. With wavelength-based routing in Figure 2 (a), for example, the OCS layer directs light from a source port, based on the input wavelength, to a given destination port. Each wavelength λ_i then implements a matching m_i between nodes, and by transmitting different wavelengths across time slots, nodes implement a schedule of matchings, that together emulate a static logical topology. Although in practice more flexibility may be available, for now, we assume that all nodes synchronously follow a given matching schedule, that is reconfigurable. Figure 2 (b) shows the set of matchings available for our example setup: each row shows a source node, and the columns show the respective destination in each matching from m_1 to m_5.

Topology. By permuting the available matchings in a schedule, we may realize several logical topologies from a given physical setup, to concentrate bandwidth in cliques of nodes. Let each node's aggregate bandwidth be b. If a

circuit appears in a fraction $l \leq 1$ of the schedule's time slots, then the circuit implements a virtual edge of bandwidth bl. For example, Figure 2 (d) shows one possible schedule of matchings, that implements logical topology A with two cliques of size 4 for our 8-node network. In this case, node bandwidth allocated within cliques is thrice that available across cliques. This supports traffic locality within cliques and uniform aggregate bandwidth across them. The setup can also realize topology B (see Figure 2 (e)) with cliques of size 2. The schedule may be further tweaked to support varying degrees of oversubscription within cliques, or non-uniform bandwidth across cliques. The supportable load depends, of course, on the routing scheme implemented, for which we present one possible choice.

Routing. To support arbitrary communication patterns within cliques, we use oblivious routing as a building block. For intra-clique traffic, we treat each clique as its own ORN and use two-hop VLB-based routing. As in existing ORN designs, the first hop is a load-balancing hop via the first available intra-clique link[1]. Inter-clique traffic uses 3 hops: first via the first available intra-clique link, then the inter-clique link to the destination clique, and finally the intra-clique link to the final destination. Again, the first hop load-balances the subsequent hops, absorbing uneven distribution of inter-clique traffic across individual source-destination pairs. In topology A for example, a flow from 0 to 6 could be routed as 0->3->7->6, or 0->1->4->6, besides other paths.

To analyze latency and throughput, we define $q \geq 1$ as the ratio between node bandwidth allocated to intra-clique and inter-clique links, i.e. the oversubscription ratio. For instance, in topology A, $q = 3$. To simplify our analysis, we assume N_c equal-sized cliques. Finally, we define $0 \leq x \leq 1$ as the fraction of total demand that is intra-clique.

Latency. We evaluate *intrinsic* latency (δ_m): the maximum number of circuits to cycle through across all hops. This is the minimum worst-case latency for a given topology and routing scheme, regardless of other deployment parameters. Our routing ensures that first hops add effectively zero intrinsic latency. For intra-clique traffic, the second hop is on the direct intra-clique link to the destination. This may require cycling through all $(\frac{N}{N_c} - 1)$ intra-clique links. Since these links only use $\frac{q}{q+1}$ of all time slots, $\delta_m = \frac{q+1}{q} \times (\frac{N}{N_c} - 1)$. For inter-clique traffic, accounting for both the inter- and intra-clique hops, $\delta_m = (q+1)(N_c - 1) + \frac{q+1}{q} \times (\frac{N}{N_c} - 1)$. Increasing oversubscription q or number of cliques N_c lowers latency for local traffic, but increases latency across cliques.

Throughput. We define the throughput $0 \leq r \leq 1$ as the fraction of total bandwidth used to deliver traffic on its final hop to the destination. This is bounded by the bandwidth available on both intra- and inter-clique links. Intra-clique links are allocated $\frac{q}{q+1}$ of total bandwidth; since all traffic must traverse these links twice, $r \leq \frac{q}{2q+2}$. Inter-clique links are allocated just $\frac{1}{q+1}$ of total bandwidth, but these are only used as direct links by the $(1-x)$ of traffic that is inter-clique. Therefore, $r \leq \frac{1}{(1-x)(q+1)}$. To maximize throughput, we set the oversubscription q such that both intra- and inter-clique links are fully utilized. By equating our two limits on r, we find that the ideal oversubscription ratio is $q = \frac{2}{1-x}$, which allows our design to support throughput $r = \frac{1}{3-x}$.

Note that r is bounded between $\frac{1}{3}$ and $\frac{1}{2}$, with higher locality ratios offering higher throughput. Figure 2 (f) shows this theoretical scaling, along with a simulation of 128 nodes and 8 cliques using real-world traffic [2]. Even with minimal locality ratios, this reduces load-balancing hops from optimal oblivious designs to increase throughput.

System	Max hops	δ_m	Min Latency	Thpt.	Norm. BW cost
Optimal ORN 1D (Sirius) [5]	2	4095	26.59 μs	50%	$2x$
Opera [18]				31.25%	$3.2x$
short flows	4	0	2 μs		
bulk	2	4095	23,034 μs		
Optimal ORN 2D [4]	4	252	3.57 μs	25%	$4x$
SORN $N_c = 64$				40.98%	$2.44x$
intra-clique	2	77	1.48 μs		
inter-clique	3	364	3.77 μs		
SORN $N_c = 32$				40.98%	$2.44x$
intra-clique	2	155	1.97 μs		
inter-clique	3	296	3.35 μs		

Table 1: A comparison of latency and throughput between existing oblivious designs, and our semi-oblivious reconfigurable network for 4096-rack DCN.

Table 1 compares our SORN proposal to existing ORN designs. We consider a DCN of 4096 racks, each with 16 uplinks, connected to Arrayed Wavelength Grating Routers (AWGR) as in Sirius [5], and assume 100 ns time slots, and 500 ns of propagation delay per hop. We remove the effects of queuing and show latency for a single packet using various systems. Opera requires longer time slots to route short flows on fixed topologies; we assume 90 μs time slots from the original paper [18], and set 1/4th of the uplinks to reconfigure at a given time. We assume a 56% locality ratio and a short flow traffic share of 75%, using median values from a production datacenter trace [23].

These results show that a semi-oblivious design uses bandwidth efficiently, to simultaneously achieve low latency and high throughput. The hierarchical structure of SORN reduces

[1]Sufficiently long flows will evenly load-balance across all intra-clique links, while variation in flow arrival times load-balances short flows.

cycle time from 1D ORNs like Sirius, in a similar way to optimal 2D ORNs, reducing total latency by an order of magnitude. For workloads in which latency within cliques is crucial, SORN outperforms both 2D ORNs and Opera due to the reduced hops. At the same time, SORN achieves throughput that nears that of 1D ORNs. In particular, by acknowledging spatial locality and aggregated demand requirements, we are able to reduce the number of indirect hops across traffic to an average of only 2.44 hops for our assumed locality ratio, utilizing bandwidth effectively.

5 Adapting the Topology

As network requirements vary with the application workload, adapting our semi-oblivious design ensures optimal performance benefits. As described earlier, coordination between a centralized network control plane and application layer entities like schedulers helps determine suitable locality and aggregate bandwidth requirements. Here, we show that coarse-grained flexibility in existing fast circuit switches allows infrequent updates to the circuit schedule to adapt to these patterns.

Consider a fast circuit-switched setup using AWGRs and fast-tunable lasers, similar to Sirius [5]. The circuit schedule for this setup is implemented entirely at the nodes themselves, by transmitting different wavelengths across time slots to enable different matchings. To update the logical topology, we can then simply update the state at each node. Based on the stability of macro-scale patterns, we expect to update the topology on the order of several minutes or hours. We could therefore use a logically centralized control plane to synchronously update state across nodes (ToR switches / end-hosts) within a few seconds [9]. An important consideration during such updates is the routing and buffering state at each node's NIC, since these functions are performed entirely at the nodes' NICs. Our semi-oblivious abstraction maintains a fixed superset of neighbors per node, varying bandwidth per neighbor. This, along with the simple, unified routing protocol, avoids new hardware state during schedule update operations, and minimizes drain operations between neighbor queues. The scope of this paper omits a full discussion, but Figure 2 (c) shows a schematic of node hardware state, which is expected to scale well with system size by leveraging efficient hardware data structures [24–26] and smarter caching between SRAM and DRAM [14].

Expressivity. As shown in the previous section, rebalancing the circuit schedule reconfigures the hierarchical topology, clustering machines in different ways. The flexibility of our framework primarily depends on two physical factors: the ports available at nodes and OCSes, and the matchings available per OCS. If the physical setup provides all $N!$ matchings, then we can choose arbitrary logical topologies. While this is not usually feasible, in practice, existing hardware can offer more than enough flexibility for DCN operators. For example, in our 4096-node network with 16 ports per node, 256-port gratings enable a circuit between each node pair, to allow clique sizes ranging from 1 (flat network) 16, 32, 64 up to 2048. Instead, we may wish to accommodate a fewer number of clique sizes, say up to 64, but allow non-uniform connectivity across these with the hundreds of remaining matchings. Further, with wavelength-selective OCSes, nodes could choose to emit different wavelengths at the same time, increasing flexibility significantly. With heterogeneity in node bandwidth and ports available, we may encode gravity models, non-uniform clique sizes, or generally allow higher provisioning between certain spatial groups. We also observe that even when equipped with full flexibility, reconfigurable networks in production prefer a fixed set of intuitive topologies [22]. Modern ToR switches with 100s of uplink ports [5] can therefore provide sufficient flexibility. An offline optimization can predetermine a set of matchings, or even schedules, for the desired level of flexibility. And routing could be chosen to accommodate highly dense, uniform all-to-all, non-uniform, or even anti-affinity patterns, between cliques.

6 Discussion and Open Questions

Our key takeaway in this paper is that reconfiguring the datacenter network to match structural patterns effectively employs node bandwidth, for high performance at low costs. The proposed techniques enable a spectrum of topologies, from clusters of high communication density, to all-to-all flat topologies, to non-uniform network requirements. This supports a diverse set of modern datacenter applications, allowing flexibility and ease of operation. For instance, consider Facebook's datacenter services described in [23]. Since each machine has a specific role, the network may be optimized for the structural relationship between different services such as user-facing web applications, cache services, and Hadoop jobs. A reconfigurable design can support both short-term fluctuation in the distribution of these services, or longer-term evolution of requirements. Even when adapting the topology is not beneficial, a structured reconfigurable network offers significant cost benefits over existing packet and circuit switching designs.

Cooperation with application-level job placement can further promote such flexibility. Importantly, a feedback loop communicating high-level application requirements to the network can easily dictate the structural properties we target, still allowing application-level entities to remain largely unrestricted and independent. We also note that our framework does not require precise predictions, maintaining guarantees within a healthy estimation error margin. Moreover, discourse on semi-oblivious designs doesn't stop here; below are some future directions that can augment our findings:

Other Structural Patterns: Non-spatial properties in the demand could also provide optimization opportunities. Diurnal utilization patterns or the distribution of latency-sensitive vs bulk traffic, for example, could help tune the number of indirect hops in reconfigurable topologies like Opera, for an adaptive latency-throughput tradeoff. Probabilistic traffic analysis [34], may also offer insights for topology design.

Machine Learning Workloads: While several works show that an optimized topology improves training/inference speeds for individual ML jobs, and that such workloads are predictable [33], at larger scales it is not always possible to arbitrarily match topology to demand. Further, in a cluster shared across training/inference jobs for different ML models, fine-grained optimization for individual jobs may be slow, or may cause fragmentation of GPU resources. Robust optimization techniques such as our semi-oblivious model, when co-designed with job scheduling, may improve resource utilization in such clusters, as we could optimize for a varying distribution of training jobs.

Practicality benefits: The adoption of reconfigurable datacenter networks faces several practical challenges. We believe that adoption of structure may also help tame some of these challenges. For instance, flat oblivious designs with many random indirect hops inflate the blast radius of failures since flows between any source-destination pair can be affected by any link/node failure. A modular design reduces this significantly and enables ease of diagnosis and reasoning. Modularity can also relax time-synchronization requirements, as a node participates in independent schedules on each hierarchical level, reducing the diameter of an individual synchronization domain. Smaller schedules may also better tolerate larger time slots and synchronization overheads.

To conclude, we believe that semi-oblivious designs apply to various contexts where coarse-grained demand information is available. We have sketched benefits and feasibility to a preliminary extent, but there is much scope for other designs and exploration in specific settings.

7 Related Work

Most existing designs for fast-circuit switched datacenters are demand-oblivious in both their topology and routing [3, 5, 20, 27]. Some systems use flow size information to route latency- and throughput-sensitive traffic separately but are otherwise oblivious [15, 18]. Wilson et al. [34] propose an oblivious circuit schedule and dynamically adjust the oblivious routing scheme for congestion, but this doesn't directly consider traffic patterns. NegotiaToR [16] augments an all-to-all circuit schedule with a demand-aware phase and uses a decentralized control plane to schedule flows in this phase. This improves performance over oblivious designs, but it is unclear if such individual flow scheduling can scale beyond a few hundred nodes.

Jupiter Evolving [22] uses slow circuit switches with arbitrary flexibility to infrequently adapt the topology to pod- or block-level traffic matrices. Researchers have proposed robust topology optimization and traffic engineering techniques for such setups [30, 37]. However, scaling circuit switching beyond the block level requires fast circuit switching to support the rapidly changing traffic patterns found at smaller scales. Since fixed schedules avoid dynamic prediction of these patterns to ensure low latency, we propose to apply infrequent demand-aware adaptations similar to prior work, to circuit schedules for fast OCSes. This brings significant bandwidth improvements while retaining low latency, and respecting the reduced flexibility found in current fast OCS designs.

Acknowledgments

We would like to thank our anonymous reviewers for their extensive and constructive feedback. This work was supported by NSF grants CHS-1955125, DBI-2019674, CNS-2331111, CAREER-2239829, CCF-2402851, and CCF-2402852.

References

[1] 2023. Mission Apollo: Behind Google's optical circuit switching revolution. https://www.datacenterdynamics.com/en/analysis/mission-apollo-behind-googles-optical-circuit-switching-revolution-mag/.

[2] Mohammad Alizadeh, Shuang Yang, Milad Sharif, Sachin Katti, Nick McKeown, Balaji Prabhakar, and Scott Shenker. 2013. pFabric: Minimal near-optimal datacenter transport. *ACM SIGCOMM Computer Communication Review* 43, 4 (2013), 435–446.

[3] Daniel Amir, Nitika Saran, Tegan Wilson, Robert Kleinberg, Vishal Shrivastav, and Hakim Weatherspoon. 2024. Shale: A Practical, Scalable Oblivious Reconfigurable Network. In *Proceedings of the ACM SIGCOMM 2024 Conference* (Sydney, NSW, Australia) *(ACM SIGCOMM '24)*. Association for Computing Machinery, New York, NY, USA, 449–464. https://doi.org/10.1145/3651890.3672248

[4] Daniel Amir, Tegan Wilson, Vishal Shrivastav, Hakim Weatherspoon, Robert Kleinberg, and Rachit Agarwal. 2022. Optimal oblivious reconfigurable networks. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing (STOC 2022)*. Association for Computing Machinery. https://doi.org/10.1145/3519935.3520020

[5] Hitesh Ballani, Paolo Costa, Raphael Behrendt, Daniel Cletheroe, Istvan Haller, Krzysztof Jozwik, Fotini Karinou, Sophie Lange, Kai Shi, Benn Thomsen, and Hugh Williams. 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* (Virtual Event, USA) *(SIGCOMM '20)*. Association for Computing Machinery, New York, NY, USA, 782–797. https://doi.org/10.1145/3387514.3406221

[6] Luiz André Barroso, Urs Hölzle, and Parthasarathy Ranganathan. 2019. *The datacenter as a computer: Designing warehouse-scale machines*. Springer Nature.

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

[8] Christina Delimitrou, Sriram Sankar, Aman Kansal, and Christos Kozyrakis. 2012. ECHO: Recreating network traffic maps for datacenters with tens of thousands of servers. In *2012 IEEE International Symposium on Workload Characterization (IISWC)*. IEEE, 14–24.

[9] Andrew D Ferguson, Steve Gribble, Chi-Yao Hong, Charles Killian, Waqar Mohsin, Henrik Muehe, Joon Ong, Leon Poutievski, Arjun Singh, Lorenzo Vicisano, et al. 2021. Orion: Google's {Software-Defined} Networking Control Plane. In *18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)*. 83–98.

[10] 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* (Florianopolis, Brazil) *(SIGCOMM '16)*. Association for Computing Machinery, New York, NY, USA, 216–229.

[11] Chuanxiong Guo, Guohan Lu, Dan Li, Haitao Wu, Xuan Zhang, Yunfeng Shi, Chen Tian, Yongguang Zhang, and Songwu Lu. 2009. BCube: a high performance, server-centric network architecture for modular data centers. In *Proceedings of the ACM SIGCOMM 2009 Conference on Data Communication* (Barcelona, Spain) *(SIGCOMM '09)*. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/1592568.1592577

[12] Ori Hadary, Luke Marshall, Ishai Menache, Abhisek Pan, Esaias E Greeff, David Dion, Star Dorminey, Shailesh Joshi, Yang Chen, Mark Russinovich, et al. 2020. Protean:{VM} allocation service at scale. In *14th USENIX Symposium on Operating Systems Design and Implementation (OSDI 20)*. 845–861.

[13] Daniel Halperin, Srikanth Kandula, Jitendra Padhye, Paramvir Bahl, and David Wetherall. 2011. Augmenting data center networks with multi-gigabit wireless links. In *Proceedings of the ACM SIGCOMM 2011 Conference (SIGCOMM '11)*. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/2018436.2018442

[14] Jason Lei and Vishal Shrivastav. 2024. Seer: Enabling Future-Aware Online Caching in Networked Systems. In *21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24)*. USENIX Association, Santa Clara, CA, 635–649. https://www.usenix.org/conference/nsdi24/presentation/lei

[15] Jialong Li, Haotian Gong, Federico De Marchi, Aoyu Gong, Yiming Lei, Wei Bai, and Yiting Xia. 2024. Uniform-Cost Multi-Path Routing for Reconfigurable Data Center Networks. In *Proceedings of the ACM SIGCOMM 2024 Conference* (Sydney, NSW, Australia) *(ACM SIGCOMM '24)*. Association for Computing Machinery, New York, NY, USA, 433–448. https://doi.org/10.1145/3651890.3672245

[16] Cong Liang, Xiangli Song, Jing Cheng, Mowei Wang, Yashe Liu, Zhenhua Liu, Shizhen Zhao, and Yong Cui. 2024. NegotiaToR: Towards A Simple Yet Effective On-demand Reconfigurable Datacenter Network. In *Proceedings of the ACM SIGCOMM 2024 Conference* (Sydney, NSW, Australia) *(ACM SIGCOMM '24)*. Association for Computing Machinery, New York, NY, USA, 415–432. https://doi.org/10.1145/3651890.3672222

[17] Hong Liu, Ryohei Urata, Kevin Yasumura, Xiang Zhou, Roy Bannon, Jill Berger, Pedram Dashti, Norm Jouppi, Cedric Lam, Sheng Li, Erji Mao, Daniel Nelson, George Papen, Mukarram Tariq, and Amin Vahdat. 2023. Lightwave Fabrics: At-Scale Optical Circuit Switching for Datacenter and Machine Learning Systems. In *Proceedings of the ACM SIGCOMM 2023 Conference*. Association for Computing Machinery, New York, NY, USA, 499–515. https://doi.org/10.1145/3603269.3604836

[18] William M. Mellette, Rajdeep Das, Yibo Guo, Rob McGuinness, Alex C. Snoeren, and George Porter. 2020. Expanding across time to deliver bandwidth efficiency and low latency. In *17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20)*. USENIX Association.

[19] W. M. Mellette, A. Forencich, J. Kelley, J. Ford, G. Porter, A. C. Snoeren, and G. Papen. 2020. Optical networking within the lightwave energy-efficient datacenter project. *Journal of Optical Communications and Networking* (2020).

[20] 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.

[21] 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. In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM* (Hong Kong, China) *(SIGCOMM '13)*. Association for Computing Machinery, New York, NY, USA, 12 pages. https://doi.org/10.1145/2486001.2486007

[22] Leon Poutievski, Omid Mashayekhi, Joon Ong, Arjun Singh, Mukarram Tariq, Rui Wang, Jianan Zhang, Virginia Beauregard, Patrick Conner, Steve Gribble, Rishi Kapoor, Stephen Kratzer, Nanfang Li, Hong Liu, Karthik Nagaraj, Jason Ornstein, Samir Sawhney, Ryohei Urata, Lorenzo Vicisano, Kevin Yasumura, Shidong Zhang, Junlan Zhou, and Amin Vahdat. 2022. Jupiter evolving: transforming google's datacenter network via optical circuit switches and software-defined networking. In *Proceedings of the ACM SIGCOMM 2022 Conference (SIGCOMM '22)*. Association for Computing Machinery.

[23] 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 (SIGCOMM '15)*. Association for Computing Machinery, New York, NY, USA.

[24] Vishal Shrivastav. 2019. Fast, Scalable, and Programmable Packet Scheduler in Hardware. In *Proceedings of the ACM Special Interest Group on Data Communication* (Beijing, China) *(SIGCOMM '19)*. Association for Computing Machinery, New York, NY, USA, 367–379. https://doi.org/10.1145/3341302.3342090

[25] Vishal Shrivastav. 2022. Programmable Multi-Dimensional Table Filters for Line Rate Network Functions. In *Proceedings of the ACM SIGCOMM 2022 Conference* (Amsterdam, Netherlands) *(SIGCOMM '22)*. Association for Computing Machinery, New York, NY, USA, 649–662. https://doi.org/10.1145/3544216.3544266

[26] Vishal Shrivastav. 2022. Stateful Multi-Pipelined Programmable Switches. In *Proceedings of the ACM SIGCOMM 2022 Conference* (Amsterdam, Netherlands) *(SIGCOMM '22)*. Association for Computing Machinery, New York, NY, USA, 663–676. https://doi.org/10.1145/3544216.3544269

[27] 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)*. USENIX Association, Boston, MA, 255–270. https://www.usenix.org/conference/nsdi19/presentation/shrivastav

[28] Arjun Singh, Joon Ong, Amit Agarwal, Glen Anderson, Ashby Armistead, Roy Bannon, Seb Boving, Gaurav Desai, Bob Felderman, Paulie Germano, Anand Kanagala, Jeff Provost, Jason Simmons, Eiichi Tanda, Jim Wanderer, Urs Hölzle, Stephen Stuart, and Amin Vahdat. 2015. Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication (SIGCOMM '15)*. Association for Computing Machinery. https://doi.org/10.1145/2785956.2787508

[29] Chunqiang Tang, Kenny Yu, Kaushik Veeraraghavan, Jonathan Kaldor, Scott Michelson, Thawan Kooburat, Aravind Anbudurai, Matthew

Clark, Kabir Gogia, Long Cheng, Ben Christensen, Alex Gartrell, Maxim Khutornenko, Sachin Kulkarni, Marcin Pawlowski, Tuomas Pelkonen, Andre Rodrigues, Rounak Tibrewal, Vaishnavi Venkatesan, and Peter Zhang. 2020. Twine: a unified cluster management system for shared infrastructure. In *Proceedings of the 14th USENIX Conference on Operating Systems Design and Implementation (OSDI'20)*. USENIX Association, USA, Article 45.

[30] Min Yee Teh, Shizhen Zhao, Peirui Cao, and Keren Bergman. 2020. COUDER: robust topology engineering for optical circuit switched data center networks. *arXiv preprint arXiv:2010.00090* (2020).

[31] Leslie G Valiant and Gordon J Brebner. 1981. Universal schemes for parallel communication. In *Proceedings of the thirteenth annual ACM symposium on Theory of computing*. 263–277.

[32] Guohui Wang, David G. Andersen, Michael Kaminsky, Konstantina Papagiannaki, T.S. Eugene Ng, Michael Kozuch, and Michael Ryan. 2010. c-Through: part-time optics in data centers. In *Proceedings of the ACM SIGCOMM 2010 Conference (SIGCOMM '10)*. Association for Computing Machinery. https://doi.org/10.1145/1851182.1851222

[33] Weiyang Wang, Moein Khazraee, Zhizhen Zhong, Manya Ghobadi, Zhihao Jia, Dheevatsa Mudigere, Ying Zhang, and Anthony Kewitsch. 2023. TopoOpt: Co-optimizing Network Topology and Parallelization Strategy for Distributed Training Jobs. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*. USENIX Association, Boston, MA, 739–767. https://www.usenix.org/conference/nsdi23/presentation/wang-weiyang

[34] Tegan Wilson, Daniel Amir, Nitika Saran, Robert Kleinberg, Vishal Shrivastav, and Hakim Weatherspoon. 2024. Breaking the VLB Barrier for Oblivious Reconfigurable Networks. In *Proceedings of the 56th Annual ACM SIGACT Symposium on Theory of Computing (STOC 2024)*. Association for Computing Machinery.

[35] Tegan Wilson, Daniel Amir, Vishal Shrivastav, Hakim Weatherspoon, and Robert Kleinberg. 2023. Extending Optimal Oblivious Reconfigurable Networks to all N. In *2023 Symposium on Algorithmic Principles of Computer Systems (APOCS)*. 1–16. https://doi.org/10.1137/1.9781611977578.ch1

[36] Johannes Zerwas, Csaba Györgyi, Andreas Blenk, Stefan Schmid, and Chen Avin. 2023. Duo: A High-Throughput Reconfigurable Datacenter Network Using Local Routing and Control. *Proc. ACM Meas. Anal. Comput. Syst.* 7, 1 (2023). https://doi.org/10.1145/3579449

[37] Mingyang Zhang, Jianan Zhang, Rui Wang, Ramesh Govindan, Jeffrey C Mogul, and Amin Vahdat. 2021. Gemini: Practical reconfigurable datacenter networks with topology and traffic engineering. *arXiv preprint arXiv:2110.08374* (2021).