



# Optimal Oblivious Reconfigurable Networks

Daniel Amir\*  
Cornell University  
Ithaca, New York, USA

Tegan Wilson\*  
Cornell University  
Ithaca, New York, USA

Vishal Shrivastav\*  
Purdue University  
West Lafayette, Indiana, USA

Hakim Weatherspoon\*  
Cornell University  
Ithaca, New York, USA

Robert Kleinberg\*  
Cornell University  
Ithaca, New York, USA

Rachit Agarwal\*  
Cornell University  
Ithaca, New York, USA

## ABSTRACT

Oblivious routing has a long history in both the theory and practice of networking. In this work we initiate the formal study of oblivious routing in the context of reconfigurable networks, a new architecture that has recently come to the fore in datacenter networking. These networks allow a rapidly changing bounded-degree pattern of interconnections between nodes, but the network topology and the selection of routing paths must both be oblivious to the traffic demand matrix. Our focus is on the trade-off between maximizing throughput and minimizing latency in these networks. For every constant throughput rate, we characterize (up to a constant factor) the minimum latency achievable by an oblivious reconfigurable network design that satisfies the given throughput guarantee. The trade-off between these two objectives turns out to be surprisingly subtle: the curve depicting it has an unexpected scalloped shape reflecting the fact that load-balancing becomes more difficult when the average length of routing paths is not an integer because equalizing all the path lengths is not possible. The proof of our lower bound uses LP duality to verify that Valiant load balancing is the most efficient oblivious routing scheme when used in combination with an optimally-designed reconfigurable network topology. The proof of our upper bound uses an algebraic construction in which the network nodes are identified with vectors over a finite field, the network topology is described by either the elementary basis or a sequence of Vandermonde matrices, and routing paths are constructed by selecting columns of these matrices to yield the appropriate mixture of path lengths within the shortest possible time interval.

## CCS CONCEPTS

• Theory of computation → Network flows; • Networks → Network design principles; Network algorithms.

\* Author order was randomized with students placed before professors.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](mailto:permissions@acm.org).

STOC '22, June 20–24, 2022, Rome, Italy

© 2022 Association for Computing Machinery.

ACM ISBN 978-1-4503-9264-8/22/06...\$15.00

<https://doi.org/10.1145/3519935.3520020>

## KEYWORDS

Oblivious routing, reconfigurable networks, valiant load balancing, circuit-switched networks

## ACM Reference Format:

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 '22)*, June 20–24, 2022, Rome, Italy. ACM, New York, NY, USA, 14 pages. <https://doi.org/10.1145/3519935.3520020>

## 1 INTRODUCTION

Oblivious routing has a long history in both the theory and practice of networking. By design, an oblivious routing scheme forwards data along a fixed path (or distribution over paths) designed to provide good performance across a wide range of possible traffic demand matrices. Past theoretical work on oblivious routing schemes focused on their ability to approximate the congestion of the optimal multicommodity flow, culminating in Räcke's discovery [22] of oblivious routing schemes for general networks that are guaranteed to approximate the optimum congestion within a logarithmic factor in the worst case. However, thus far, oblivious routing has only been studied in the context of static networks, where the edges in the network are fixed at the beginning and do not change over time. Recent works in datacenter network architecture [8, 9, 17–20, 23, 27] have advanced *reconfigurable networks* as a promising direction. A reconfigurable network is defined as a  $d$ -regular network with  $N$  nodes (or hosts) where the edges (or links) between the nodes can be reconfigured (or rearranged) very rapidly over time. Early designs of reconfigurable networks for datacenters [8, 17, 27] relied on predictable traffic demand matrices to choose optimal edge configurations and routes for sending data between nodes. However, more recent works [18, 19, 23] in this space have made a case that traffic demand matrices in datacenters are highly unpredictable and change at very fine time granularities, making it challenging, if not impossible, to accurately track the demand matrix at any given time. To overcome this fundamental challenge, recent works have advocated for edge configuration and route selection mechanisms that are *oblivious* to traffic demand matrices. In this paper, we make the first attempt to formally study the problem of oblivious routing in the novel context of reconfigurable networks.

There are two key objectives that oblivious reconfigurable networks must aim to optimize. First, since it is costly to overprovision networks (especially for modern high-bandwidth links), datacenter network operators aim for extremely high throughput, utilizing a large *constant* factor of the available network capacity at all times

if possible. At the same time, it is desirable to minimize latency, the worst-case delay between when a packet arrives to the network and when it reaches its destination. Thus, there is a vital need to understand oblivious network designs for reconfigurable networks that guarantee high throughput and low maximum latency.

The objectives of maximizing throughput and minimizing latency in reconfigurable networks are in conflict: due to degree constraints most nodes cannot be connected by a direct link at all times, so one has to either use indirect paths, which comes at the expense of throughput, or settle for higher latency while waiting for reconfigurations to yield a more direct path. Since different deployments (and applications) may necessitate different tradeoffs between these two conflicting objectives, the main question that our work investigates is the following:

*For every throughput rate  $r$ , what is the minimum latency achievable by an oblivious reconfigurable network design that guarantees throughput  $r$ ?*

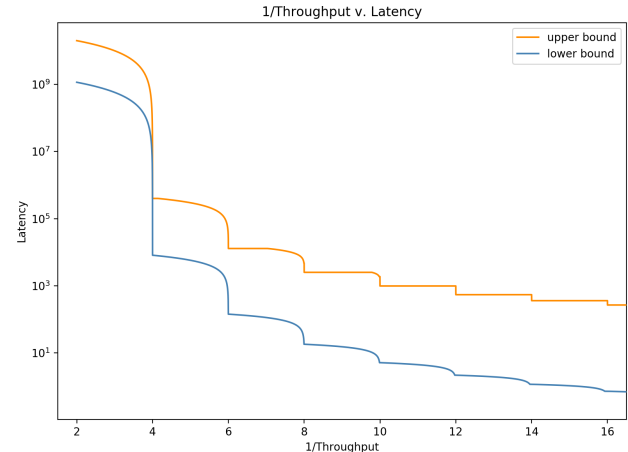
We fully resolve this question to within a constant factor<sup>1</sup> for  $d$ -regular reconfigurable networks, except when  $d$  is very large — bounded below by a constant power of  $N$  (the number of nodes in the network). That is, for every constant rate  $r$ , we identify a lower bound  $\frac{1}{d}L^*(r, N)$  such that any  $N$ -node  $d$ -regular reconfigurable network guaranteeing throughput  $r$  must have maximum latency bounded below by  $\frac{1}{d}L^*(r, N)$ . Complementing this lower bound, we design oblivious networking schemes that guarantee throughput  $r$  and have maximum latency bounded by  $O(\frac{1}{d}L^*(r, N))$ , for every constant  $r \in (0, \frac{1}{2}]$ ,  $d \in \mathbb{N}$ , and infinitely many  $N$ . (For  $r > \frac{1}{2} + o(1)$ , we show in the full version of this paper [1] that it is impossible for oblivious network designs to guarantee throughput  $r$ .)

The shape of the optimal tradeoff curve between throughput and latency is quite surprising. Figure 1 depicts the curve for  $N = 10^9$  and  $d = 1$ ; the  $x$ -axis measures the inverse throughput,  $1/r$ , while the  $y$ -axis (in log scale) measures maximum latency. The curve is scallop-shaped, with particularly favorable tradeoffs occurring when  $1/r$  is an even integer. Between even-integer values of  $1/r$ , the maximum latency improves slowly at first, then precipitously as  $1/r$  approaches the next even integer. The proof of our main result explains these key features of the tradeoff curve: its non-convexity, the special role played by even integer values of  $1/r$ , and the steep but continuous improvement in  $L^*(r, N, d)$  as  $1/r$  approaches the next even integer. In Section 1.2 below we sketch the intuitions that account for these features. Before doing so, we pause to explain more fully our model and notation.

## 1.1 Our Model and Results

Our model of oblivious reconfigurable networking is inspired by the circuit-switched network designs popularized by works such

<sup>1</sup> One could, of course, ask the transposed question: *for every latency bound  $L$ , what is the maximum guaranteed throughput rate achievable by an oblivious routing scheme with maximum latency  $L$ ?* Our work also resolves this question, not only to within a constant factor, but up to an additive error that tends to zero as  $N \rightarrow \infty$ . As noted below in Section 1.2, optimizing throughput to within a factor of two, subject to a latency bound, is much easier than optimizing latency to within a constant factor subject to a throughput bound. The importance of the latter optimization problem, *i.e.* our main question, is justified by the high cost of overprovisioning networks: due to the cost of overprovisioning, datacenter network operators tend to be much less tolerant of suboptimal throughput than of suboptimal latency.



**Figure 1:** A plot of the upper and lower bounds for the latency of an ORN containing  $10^9$  nodes that can guarantee a given throughput.

as [18, 19, 23]. These are networks composed of a fixed set of  $N$  nodes, with a switching fabric that allows a time-varying pattern of links providing connectivity between node pairs. A network design in our model is specified by two ingredients: a connection schedule and an oblivious routing scheme. The connection schedule designates which node pairs are connected in each timeslot. This can be visualized in the form of a virtual topology: a layered directed graph (with layers corresponding to timeslots) that encodes the paths that network traffic can take over time. The oblivious routing scheme designates, for each source-destination pair  $(a, b)$  and timeslot  $t$ , a probability distribution over routing paths used to forward traffic with destination  $b$  that originates at  $a$  in timeslot  $t$ . A routing path is specified by the sequence of edges in the virtual topology that compose the path. We call the combination of a connection schedule and an oblivious routing scheme an *oblivious reconfigurable network (ORN) design*.

We evaluate ORN designs according to two quantities: maximum latency ( $L$ ) and guaranteed throughput ( $r$ ). Latency of a path measures the difference between the timeslots when it starts and ends, and an ORN design with maximum latency  $L$  uses no routing paths of latency greater than  $L$ . The definition of guaranteed throughput is more subtle. First, we model demand using a function that specifies, for each source-destination pair and each timeslot, the amount of flow with that source and destination originating at that time. We say an ORN design *guarantees throughput  $r$*  if the routing scheme is guaranteed not to exceed the capacity of any link, whenever the demand satisfies the property that the total amount of demand originating at any source, or bound for any destination, never exceeds  $r$  at any timeslot. Our main result can now be stated in the following form.

**Theorem 1.** *Consider any constant  $r \in (0, \frac{1}{2}]$ . Let  $(h, \varepsilon)$  to be the unique solution in  $\mathbb{N} \times (0, 1]$  to the equation  $\frac{1}{2r} = h + 1 - \varepsilon$ , and let  $L^*(r, N)$  be the function*

$$L^*(r, N) = h \left( N^{1/(h+1)} + (\varepsilon N)^{1/h} \right).$$

For every  $N > 1$  and every ORN design on  $N$  nodes that guarantees throughput  $r$ , the maximum latency is at least  $\Omega(L^*(r, N))$ . Furthermore for infinitely many  $N$  there exists an ORN design on  $N$  nodes that guarantees throughput  $r$  and whose maximum latency is  $O(L^*(r, N))$ .

## 1.2 Techniques

To begin reasoning about the latency-throughput tradeoff in ORNs, note that for any node in the virtual topology, the number of distinct routing paths originating at that node whose latency is at most  $L$  and which contain  $p$  physical edges is  $\binom{L}{p}$ . Hence, in order for a node to be able to reach a majority of other nodes within  $L$  timeslots using at most  $h$  physical links, we must have the inequality  $\sum_{p=0}^h \binom{L}{p} \geq N/2$ . A simple calculation verifies that this inequality implies  $L = \Omega(hN^{1/h})$ . A routing scheme in which the routing path between a random source and a random destination contains  $h$  physical links, on average, cannot guarantee throughput greater than  $1/h$ . This suggests a latency-throughput relationship of the form  $L = \Omega\left(\frac{1}{r}N^r\right)$ . This lower bound can be made rigorous with a little bit of work, but it differs from the tight bound asserted in Theorem 1 in two significant ways.

- (1) Whereas  $\frac{1}{r}N^r$  is a smooth convex function of  $r > 0$ , the function  $L^*(r, N)$  is non-smooth and non-convex; when plotted as a function of  $1/r$  it exhibits a scalloped shape with cusps at even integer values of  $1/r$ .
- (2) The exponent of  $N$  in the function  $L^*(r, N)$  is approximately  $2r$  rather than  $r$ . In other words, the naïve bound  $L \geq \frac{1}{r}N^r$  is tight up to a factor of 2 in terms of throughput, but off by a factor of about  $N^r$  in terms of latency. (As remarked in Footnote 1, sacrificing a factor of 2 in throughput is typically regarded by network operators as much more costly than sacrificing a constant factor in latency.)

The first of these differences is explained by a refinement of the counting argument at the start of this section. In order to guarantee throughput  $r$ , the average number of physical hops on the routing paths used (under any traffic demands with at most  $r$  units of flow based at any source or destination) must be at most  $1/r$ . However, the number of physical hops in any path must be an integer. Thus, if  $1/r$  is not an integer, at least a constant fraction of routing paths must have  $\lfloor 1/r \rfloor$  physical hops or fewer. Subject to any upper bound on latency, paths with a limited number of physical hops are much less numerous than those with a larger number of physical hops, so the requirement to use a large number of distinct paths with  $\lfloor 1/r \rfloor$  or fewer physical hops places a significantly stricter lower bound on maximum latency, leading to the non-convex shape with regularly spaced cusps depicted in Figure 1.

To give intuition for the factor-two difference in throughput between the naïve lower bound and the true function  $L^*(r, N)$ , it is useful to recall *Valiant load balancing (VLB)*, an ingredient in many of the earliest and most practical oblivious routing schemes. VLB constructs a random path from source  $s$  to destination  $t$  by choosing a random intermediate node,  $r$ , and concatenating minimum-cost paths from  $s$  to  $r$  and from  $r$  to  $t$ . This inflates the number of physical hops used in routing paths by a factor of two, but is beneficial because it prevents congestion under worst-case demands. The fact

that the exponent of  $N$  in  $L^*(r, N)$  is approximately  $2r$  rather than  $r$  can be interpreted as confirming that the factor-two inflation due to VLB is unavoidable, for oblivious routing schemes that guarantee throughput  $r$ . To prove this fact, we formulate optimal oblivious routing for a given virtual topology as a linear program and interpret the dual variables as endpoint-specific edge costs that can be summed to ascribe a cost to every path connecting a given pair of endpoints. We prove that, regardless of the virtual topology, one can always design a carefully-constructed dual solution that penalizes paths containing a large number of physical hops, and doubly penalizes physical hops that are too close to both endpoints. Paths that avoid the double penalty must use twice as many physical hops as minimum-cost paths, exactly as in VLB routing. The most delicate part of the proof is the verification that the dual solution is feasible, which requires carefully bounding the number of nodes reachable from any source within a given cost budget.

To prove that the lower bound  $L^*(r, N)$  is tight, we need to construct an ORN design that matches the bound up to a constant factor. Our design is easiest to describe when  $r = \frac{1}{2h}$  and  $N = n^h$  for positive integer  $h$  and prime number  $n$ . In that case, we use a design that we call the *Elementary Basis Scheme (EBS)* which identifies the set of  $N$  nodes with elements of the group<sup>2</sup>  $(\mathbb{Z}/(n))^h$ . Let  $\mathbf{e}$  be the elementary basis consisting of the columns of the  $h \times h$  identity matrix. EBS uses a connection schedule whose timeslots cycle through the nonzero scalar multiples of elements of  $Y$ . In a timeslot devoted to  $s \cdot \mathbf{e}_i$ , the network is configured to allow each node  $x$  to send to  $x + s \cdot \mathbf{e}_i$ . Over the course of one complete cycle, any two nodes can be connected by a “direct path” consisting of  $h$  physical hops (or fewer) that modify the coordinates of the source node one by one until they match the coordinates of the destination. The EBS routing scheme constructs a random path connecting a given source and destination using VLB: it chooses a random intermediate node and concatenates two “semi-paths”: the direct paths from the source to the intermediate node and from the intermediate node to the destination.

To generalize this design to all non-integer values of  $\frac{1}{2r}$ , we need to enhance EBS so that a constant fraction of semi-paths use  $h$  physical hops and a constant fraction use  $h + 1$  physical hops. This necessitates a modified ORN design that we call the *Vandermonde Basis Scheme (VBS)*. Assume  $r = h + 1 - \varepsilon$  for  $h \in \mathbb{N}$ ,  $0 < \varepsilon < 1$ , and that  $N = n^{h+1}$  for prime  $n$ , so that the nodes can be identified with the vector space  $\mathbb{F}_n^{h+1}$ . Instead of one basis corresponding to the identity matrix, we now use a sequence of distinct bases each corresponding to a different Vandermonde matrix. In addition to the single-basis semi-paths (which now constitute  $h + 1$  physical hops), this enables the creation of “hop-efficient” semi-paths composed of  $h$  physical hops belonging to two or more of the Vandermonde matrices in the sequence. Hop-efficient semi-paths have higher latency than direct paths, but we opportunistically use only the ones with lowest latency to connect a subset of terminal pairs, joining the remaining pairs with direct semi-paths. A full routing path is then defined to be the concatenation of two random semi-paths, as before. Proving that the routing scheme guarantees throughput  $r$

<sup>2</sup>This should be thought of as the  $h$ -dimensional vector space over  $\mathbb{Z}/(n)$ . While we describe taking the elementary basis for simplicity here, the EBS scheme itself does not require  $\mathbb{Z}/(n)$  to be a field, thus we use the word group here. This is further described in Section 4.1.

boils down to quantifying, for each physical edge  $e$ , the net effect of shifting load from direct paths that use  $e$  to hop-efficient paths that avoid  $e$  and vice-versa. The relevant sets of paths in this calculation can be parameterized by unions of affine subspaces of  $\mathbb{F}_n^{h+1}$ , and the use of Vandermonde matrices in the connection schedule gives us control over the dimensions of intersections of these subspaces, and thus over the size of their union.

### 1.3 Related work

**Oblivious routing in general networks:** Räcke’s seminal 2002 paper [21] proved the existence of  $\text{polylog}(n)$ -competitive oblivious routing schemes in general networks. Subsequent work improved the competitive ratio [12] and devised polynomial-time algorithms for computing an oblivious routing scheme that meets this bound [3, 5, 12]. Räcke’s 2008 paper [22] yielded an  $O(\log n)$ -competitive oblivious routing scheme, computed by a fast, simple algorithm based on multiplicative weights and FRT’s randomized approximation of general metric spaces by tree metrics [7]. The effectiveness of Räcke’s 2008 routing scheme for wide-area traffic engineering in practice was demonstrated in [2, 16]. Additionally, Gupta, Hajiaghayi, and Räcke [10] show a  $\text{polylog}(n)$  competitive ratio for routing schemes oblivious to both traffic and the cost functions associated with each edge. While these works achieve excellent congestion minimization over general networks, they do not specifically consider throughput or latency, and do not attempt to co-design the network with their routing scheme.

With respect to bounding the throughput of oblivious routing schemes, Hajiaghayi, Kleinberg, Leighton, and Räcke [11] prove a lower bound of  $\Omega(\frac{\log n}{\log \log n})$  on the competitive ratio in general networks. However, their definition of throughput differs from ours; they simply mean the combined flow rate delivered to all sender-receiver pairs. With respect to latency, the competitive ratio of average latency of oblivious routing over general networks is analyzed by [13]. Their model of latency differs from ours; they assign resistance values to each edge, and they only provide an oblivious routing scheme achieving the  $O(\log(N))$ -competitive ratio when routing to a single target.

**Valiant load balancing in hypercubes and other architectures:** Leslie Valiant introduced oblivious routing in [25]. The VLB scheme for randomized routing in the hypercube was introduced, and shown to be optimal, by Valiant and Brebner [25, 26]. While these works evaluate latency under queuing, they do not evaluate throughput. Additionally, they use a direct-connect torus topology. Our work can be interpreted as proving that VLB is the optimal oblivious routing scheme to use in conjunction with an optimally-designed reconfigurable network topology, thus providing further theoretical justification for the widespread usage of VLB in practice when oblivious routing is applied on handcrafted network topologies.

Our first ORN design, EBS, for the case where  $N = 2^h$ , can be seen as simulating VLB over a hypercube. In this case, the connection schedule simulates an  $h$ -dimensional hypercube by cycling through  $h$  different connection graphs, each with degree 1. While the latency bound achieved by EBS in this case is  $O(\log(N))$ , the same bound found in [25], the differing contexts makes these bounds

incomparable. In particular, while Valiant considered queuing delay, we consider latency intrinsic to the scheduled nature of ORNs.

A lower bound for *deterministic* oblivious routing in  $d$ -regular networks with  $N$  nodes was proven in [14]; the same paper shows this bound is tight for hypercube networks, in which  $d = \log(N)$ .

**Load-Balanced Switches:** The load-balanced switch architecture proposed by Chang [6] uses static schedules and sends traffic obliviously via intermediate nodes. While there are significant similarities between this architecture and ORNs, it differs in its use of specialized intermediate nodes (rather than sending traffic via multiple end-hosts), as well as its focus on monolithic switches.

#### Circuit-Switched Datacenter Network Architectures:

c-Through [27] and Traffic Matrix Scheduling [20], as well as many other designs, propose a hybrid network in which a packet-switched backbone exists alongside a circuit-switched fabric. However, with advances in circuit switches that have reduced reconfiguration times to nanosecond-scale, it is worth reconsidering whether a separate packet-switched backbone is truly necessary.

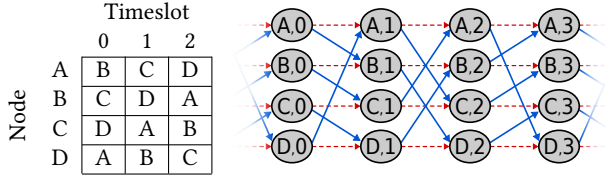
**Oblivious Circuit-Switched Networks:** Rotornet and Sirius [4, 19] are two ORN concepts proposed for datacenter-wide networks that use optical circuit switches to build a reconfigurable network fabric. Shoal [23] is a similar ORN concept that uses electric circuit switches in a disaggregated rack environment. Together, these works demonstrate that the ORN paradigm is feasible in practice. These designs use similar schedules that prioritize achieving high throughput at the expense of poor latency for large  $N$ . Our first ORN design, EBS, generalizes these existing designs to achieve many potential tradeoffs, ranging from the existing tradeoff to that achieved by an ORN version of hypercube routing.

Opera [18] evolves on the ORN concept by greatly lengthening each timeslot and creating an expander graph topology between nodes during each timeslot. Opera uses a non-oblivious routing scheme in which latency-sensitive traffic is sent via multiple hops within a single expander graph topology, while throughput-sensitive traffic is held until the schedule advances to a topology in which it can be sent directly to the destination in one hop. This design makes strong assumptions about the workload, including that bandwidth-sensitive traffic is near all-to-all, limiting its flexibility.

## 2 DEFINITIONS

This section presents definitions that formalize the notion of an oblivious reconfigurable network (ORN). We assume a network of  $N$  nodes communicating in discrete, synchronous timeslots. The nodes are joined by a communication medium that allows an arbitrary pattern of unidirectional communication links to be established in each timeslot, subject to a degree constraint that each node participates as the sender in at most  $d$  connections, and as the receiver in at most  $d$  connections. Throughout most of this paper we specialize to the case  $d = 1$ ; see Section 2.1 below for a discussion of why the general case reduces to this special case.

In systems that instantiate reconfigurable networking, data is encapsulated in fixed-size units called *frames* or *packets*. In this work we instead treat data as a continuously-divisible commodity, and we allow sending fractional quantities of flow along multiple paths from the source to the destination. This abstraction is standard in theoretical works on oblivious routing, and can be justified



**Figure 2: A connection schedule among four nodes, as well as part of its corresponding virtual topology. The full virtual topology represents a countably infinite number of timeslots.**

by interpreting fractional flow as a probability distribution over routing paths, with each frame being sent along one path sampled at random from the distribution. Under this interpretation flow values represent the expected number of frames traversing a link.

**Definition 1.** A *connection schedule*  $\pi$  with size  $N$  is a sequence of permutations  $\pi_j$  for all  $j \in \mathbb{Z}$ , each mapping  $[N]$  to  $[N]$ . The interpretation of the relation  $\pi_t(i) = j$  is that node  $i$  is allowed to send one frame to node  $j$  during timeslot  $t$ .

For the rest of this paper we will focus on *periodic* connection schedules, those with finite description size. These have some period length  $T$  such that  $\pi_t(i) = \pi_k(i)$  for all  $t \equiv k \pmod{T}$ . See the full paper [1] to see how our results extend to aperiodic schedules.

The *virtual topology* of the connection schedule  $\pi$  is a directed graph  $G_\pi$  with vertex set  $[N] \times \mathbb{Z}$ . The edge set of  $G_\pi$  consists of the union of  $E_{\text{virt}}$  and  $E_{\text{phys}}$ .  $E_{\text{virt}}$  is the set of *virtual edges*, which are of the form  $(i, t) \rightarrow (i, t+1)$  and represent the frame waiting at node  $i$  during the timeslot  $t$ .  $E_{\text{phys}}$  is the set of *physical edges*, which are of the form  $(i, t) \rightarrow (\pi_t(i), t+1)$  and represent the frame being transmitted from  $i$  to  $\pi_t(i)$  at timeslot  $t$ .

We interpret a path in  $G_\pi$  from  $(a, t)$  to  $(b, t')$  as a potential way to transmit a frame from node  $a$  to node  $b$ , beginning at timeslot  $t$  and ending at some timeslot  $t'$ . For a node  $a \in [N]$  let  $\llbracket a \rrbracket$  denote the set  $\{a\} \times \mathbb{Z}$ , consisting of all copies of  $a$  in  $G_\pi$ . Let  $\mathcal{P}(a, b, t)$  denote the set of paths in  $G_\pi$  from the vertex  $(a, t)$  to  $\llbracket b \rrbracket$ . Finally, let  $\mathcal{P} = \bigcup_{a,b,t} \mathcal{P}(a, b, t)$  denote the set of all paths in  $G_\pi$ .

**Definition 2.** A *flow* is a function  $f : \mathcal{P} \rightarrow [0, \infty)$ . For a given flow  $f$ , the amount of flow traversing an edge  $e$  is defined as:

$$F(f, e) = \sum_{P \in \mathcal{P}} f(P) \cdot \mathbf{1}_{e \in P}$$

We say that  $f$  is *feasible* if for every physical edge  $e \in E_{\text{phys}}$ ,  $F(f, e) \leq 1$ .

**Definition 3.** The *latency*  $L(P)$  of a path  $P$  in  $G_\pi$  is equal to the number of edges it contains (both virtual and physical). Note that traversing any edge in the virtual topology (either virtual or physical) is equivalent to advancing in time by the duration of one timeslot, so the number of edges in a path is proportional to the elapsed time. For a nonzero flow  $f$ , the *maximum latency* is the maximum over all paths in the flow

$$L_{\max}(f) = \max_{P \in \mathcal{P}} \{L(P) : f(P) > 0\}$$

Our definitions of latency and of the virtual topology  $G_\pi$  incorporate the idealized assumption of zero propagation delay. In other

words, we assume that a frame sent in one timeslot is received by the beginning of the following timeslot, and that the length of a path in the virtual topology accurately reflects the time interval between when the frame originates and when it reaches its destination.

**Definition 4.** An *oblivious routing scheme*  $R$  is a function that associates to every  $(a, b, t) \in [N] \times [N] \times \mathbb{Z}$  a flow  $R_{a,b,t}$  such that:

- (1)  $R_{a,b,t}$  is supported on paths from  $(a, t)$  to  $\llbracket b \rrbracket$ , meaning  $\forall P \notin \mathcal{P}(a, b, t) \ R_{a,b,t}(P) = 0$ .
- (2)  $R_{a,b,t}$  routes one unit of flow, meaning  $\sum_P R_{a,b,t}(P) = 1$ .
- (3)  $R$  has period  $T$ . In other words,  $R_{a,b,t+T}$  is equivalent to  $R_{a,b,t}$  (except with all paths transposed by  $T$  timeslots, as required to satisfy point 1).

**Definition 5.** A *demand matrix* is an  $N \times N$  matrix which associates to each ordered pair  $(a, b)$  an amount of flow to be sent from  $a$  to  $b$ . A *demand function*  $D$  is a function that associates to every  $t \in \mathbb{Z}$  a demand matrix  $D(t)$  representing the amount of flow  $D(t, a, b)$  to originate between each source-destination pair  $(a, b)$  at timeslot  $t$ . The *throughput requested by demand function*  $D$  is the maximum, over all  $t$ , of the maximum row or column sum of  $D(t)$ .

**Definition 6.** For a given oblivious routing scheme  $R$  and demand function  $D$ , the *induced flow*  $f(R, D)$  is defined by:

$$f(R, D) = \sum_{(a,b,t) \in [N] \times [N] \times \mathbb{Z}} D(t, a, b) R_{a,b,t}$$

**Definition 7.** An oblivious routing scheme is said to *guarantee throughput*  $r$  if the induced flow  $f(R, D)$  is feasible whenever the demand function  $D$  requests throughput at most  $r$ .

Definition 7 can be interpreted as meaning that the network is able to simulate a “big switch” with  $N$  input and output ports having line rate  $r$ : as long as the amount of data originating at any node  $a$  or destined for any node  $b$  does not exceed rate  $r$  per timeslot, the network is able to route all data to its destination without violating capacity constraints.

In this work, we examine the tradeoffs between guaranteed throughput and maximum latency. Specifically, among ORNs of size  $N$  that guarantee throughput  $r$ , what is the lowest possible maximum latency?

## 2.1 Allowing degree $d > 1$ in a timeslot

Although our formalization of ORNs only describes networks in which nodes have a degree of 1 in every timeslot, it can be generalized to networks that support a  $d$ -regular connectivity pattern in each timeslot. When  $d > 1$ , we interpret a demand matrix  $D$  which requests throughput  $r$  as one in which the row and column sums of  $D$  are bounded above by  $dr$ .

The connectivity of  $N \times \{t, t+1\}$  is  $d$ -regular bipartite. By König’s Theorem, this edge set can be decomposed into  $d$  edge-disjoint perfect matchings, which we use to “unroll” into  $d$  consecutive timeslots of a 1-regular ORN. Therefore, a  $d$ -regular ORN design which guarantees throughput  $r$  with maximum latency  $L$  unrolls into a 1-regular ORN design which guarantees throughput  $r$  with maximum latency  $dL$ .

Under this framework, a lower bound  $L^*(r, N)$  for 1-regular ORN designs trivially implies the lower bound  $\frac{1}{d}L^*(r, N)$  for  $d$ -regular designs. However, an upper bound for 1-regular designs

does not necessarily imply a similar upper bound for  $d$ -regular designs, because the routing scheme could route paths containing two or more physical edges in timeslots belonging to the same “unrolled” segment of the 1-regular virtual topology. This would correspond to traversing two or more edges at once in the  $d$ -regular topology. Our upper bound construction can be modified to avoid this problem. Specifically, it can be modified to never allow flow to be routed along two edges within any block of  $d$  consecutive time slots, provided  $d \leq N^{1/(h+1)}$ . This modification adds a factor of at most 2 to the maximum latency. Then, by inverting the unrolling process, we obtain a  $d$ -regular ORN design with maximum latency  $L = O(\frac{1}{d}L^*(r, N))$ . This confirms that the tight bound on maximum latency for  $d$ -regular ORN designs is  $\Theta(\frac{1}{d}L^*(r, N))$  whenever  $d \leq N^{1/(h+1)}$  and justifies our focus on the case  $d = 1$  throughout the remainder of this paper. These modifications are further described in the full version of this paper [1].

### 3 LOWER BOUND

In this section we prove the lower-bound half of Theorem 1, which says that when  $\frac{1}{2r} = h + 1 - \varepsilon$  with  $h \in \mathbb{N}$  and  $0 < \varepsilon \leq 1$ , any  $d$ -regular,  $N$ -node ORN design that guarantees throughput  $r$  must have maximum latency  $\Omega(\frac{h}{d}[N^{1/(h+1)} + (\varepsilon N)^{1/h}])$ . As noted in Section 2.1, the general case of this lower bound reduces to the case  $d = 1$ ; we assume  $d = 1$  throughout the remainder of this section.

Because the full proof is somewhat long, we begin by sketching some of the main ideas in the proof, beginning with a much simpler argument leading to a lower bound of the form  $\Omega(\frac{1}{r}N^r)$  when  $1/r$  is an integer. This simple lower bound applies not only to oblivious routing schemes, but to *any* feasible flow  $f$  that solves the uniform multicommodity flow problem given by the demand function  $D(t, a, b) = \frac{r}{N-1}$  for all  $t \in [T]$  and  $b \neq a$ . The lower bound follows by combining a few key observations.

- (1) Define the cost of a path to be the number of physical edges it contains. Since every source sends out  $r$  units of flow at all times, the flow  $f$  sends out  $rNT$  units of flow per  $T$ -step period, in a network whose physical edges have only  $NT$  units of capacity per  $T$ -step period. Consequently the average cost of flow paths in  $f$  must be at most  $\frac{1}{r}$ .
- (2) For any source node  $(a, t)$  in the virtual topology, the number of distinct destinations  $[[b]]$  that can be reached via a path with maximum latency  $L$  and cost  $p$  is bounded above by  $\binom{L}{p}$ .
- (3) If  $L \leq \frac{1}{2\varepsilon r}N^r$ , we have  $\binom{L}{1/r} \leq N/4$  and  $\sum_{p=1}^{1/r} \binom{L}{p} \leq N/2$ , so the majority of source-destination pairs cannot be joined by a path with latency  $L$  and cost less than  $\frac{1}{r} + 1$ . In fact, even if we connect every source and destination with a minimum-cost path (subject to latency bound  $L$ ), one can show that the average cost of paths will exceed  $\frac{1}{r}$ .
- (4) Since a feasible flow must have average path cost at most  $\frac{1}{r}$ , we can conclude that a feasible flow does not exist when  $L \leq \frac{1}{2\varepsilon r}N^r$ .

When  $1/r$  is an integer, this lower bound of  $L_{\max} \geq \frac{1}{2\varepsilon r}N^r$  for feasible uniform multicommodity flows turns out to be tight up to a constant factor. However for oblivious routing schemes, Theorem 1 shows that maximum latency is bounded below by a

function in which the exponent of  $N$  is roughly twice as large. Stated differently, for a given maximum latency bound, the optimal throughput guarantee for oblivious routing is only half as large as the throughput of an optimal uniform multicommodity flow.

The factor-two difference in throughput between oblivious routing and optimal uniformly multicommodity flow solutions aligns with the intuition that oblivious routing schemes must use indirect paths (as in Valiant load balancing) if they are to guarantee throughput  $r$ , whereas uniform multicommodity flow solutions (in a well-designed virtual topology) can afford to satisfy all demands using shortest-path routing. The proof of the lower bound for oblivious routing needs to substantiate this intuition.

To do so, we formulate oblivious routing as a linear program and interpret the dual variables as specifying a more refined way to measure the cost of paths. Rather than defining the cost of a path to be its number of physical edges, the duality-based proof amounts to an accounting system in which the cost of using an edge depends on the endpoints of the path in which the edge is being used. For a parameter  $\theta$  which we will set to  $h + 1$  (unless  $\varepsilon$  is very small, in which case we'll set  $\theta = h + 2$ ), the dual accounting system assesses the cost of an edge to be 1 if its distance from the source is less than  $\theta$ , plus 1 if its distance from the destination is less than  $\theta$ . Thus, the cost of an edge is doubled when it is close to both the source and the destination. The doubling has the effect of equalizing the costs of direct and indirect paths: when the distance between a source and destination is at least  $\theta$ , there is no difference in cost between a shortest path and one that combines two semi-paths each composed of  $\theta$  physical edges.

Viewed in this way, it is intuitive that the proof manages to show that VLB routing schemes, which construct routing paths by concatenating random semi-paths with the appropriate number of physical edges, correspond to optimal solutions of the oblivious routing LP. The difficulty in the proof lies in showing that the constructed dual solution is feasible; for this, we make use of a version of the same counting argument sketched above, that bounds the number of distinct destinations reachable from a given source under constraints on the maximum latency and the maximum number of physical edges used.

#### 3.1 Lower Bound Theorem Proof

Before presenting the proof of Theorem 2, we formalize the counting argument we reasoned about in our proof sketch.

**Lemma 1. (Counting Lemma)** *If in an ORN topology, some node  $a$  can reach  $k$  other nodes in at most  $L$  timeslots using at most  $h$  physical hops per path for some integer  $h$ , then  $k \leq 2\binom{L}{h}$ , assuming  $h \leq \frac{1}{3}L$ .*

**PROOF.** If node  $a$  can reach  $k$  other nodes in  $\leq L$  timeslots using exactly  $h$  physical hops per path, then  $k \leq \binom{L}{h}$ . Additionally, the function  $\binom{L}{h}$  grows at least exponentially in base 2 – that is,  $\binom{L}{h} \geq 2\binom{L}{h-1}$  – up until  $h = \frac{1}{3}L$ . Therefore, the number of such  $k$  is at most  $\sum_{i=1}^h \binom{L}{i} \leq 2\binom{L}{h}$ .  $\square$

**Theorem 2.** *Given an ORN design  $\mathcal{R}$  which guarantees throughput  $r$ , the maximum latency suffered by any routing path  $P$  with  $R_{a,b,t}(P) >$*

0 over all  $a, b, t$  is bounded by the following equation

$$L_{max} \geq \Omega \left( h \left[ (\varepsilon N)^{1/h} + N^{1/(h+1)} \right] \right) \quad (1)$$

where  $h = \lfloor \frac{1}{2r} \rfloor$  and  $\varepsilon \in (0, 1]$  is set to equal  $h+1 - \frac{1}{2r}$ . In other words,  $(h, \varepsilon)$  is the unique solution in  $\mathbb{N} \times (0, 1]$  to the equation  $\frac{1}{2r} = h+1 - \varepsilon$ .

**PROOF.** Consider the linear program below which maximizes throughput given a maximum latency constraint,  $L$ , where we let  $\mathcal{P}_L(a, b, t)$  be the set of paths from  $(a, t) \rightarrow \llbracket b \rrbracket$  with latency at most  $L$ .

**LP**

maximize:  $r$

subject to:  $\sum_{P \in \mathcal{P}_L(a, b, t)} R_{a, b, t}(P) = r$   
 $\forall a, b \in [N], t \in [T]$

$$\sum_{a \in [N]} \sum_{t=0}^{T-1} \sum_{P \in \mathcal{P}_L(a, \sigma(a), t): e \in P} R_{a, \sigma(a), t}(P) \leq 1$$

$$\forall \sigma \in S_N, e \in E_{\text{phys}}$$

$$R_{a, b, t}(P) \geq 0$$

$$\forall a, b \in [N], t \in [T], P \in \mathcal{P}_L(a, b, t)$$

The second set of constraints, in which the parameter  $\sigma$  ranges over the set  $S_N$  of all permutations of  $[N]$ , can be reformulated as the following set of nonlinear constraints in which the maximum is again taken over all permutations  $\sigma$ :

$$\max_{\sigma} \left\{ r \sum_{a \in [N]} \sum_{t=0}^{T-1} \sum_{P \in \mathcal{P}_L(a, \sigma(a), t): e \in P} R_{a, \sigma(a), t}(P) \right\} \leq 1 \quad \forall e \in E_{\text{phys}}$$

Note that given an edge  $e$ , this maximization over permutations  $\sigma$  corresponds to maximizing over perfect bipartite matchings with edge weights defined by  $w_{a, b, e} = \sum_{t=0}^{T-1} \sum_{P \in \mathcal{P}_L(a, b, t): e \in P} R_{a, b, t}(P)$ . This prompts the use of a maximum matching LP and its dual. We substitute finding a feasible matching dual solution into the original LP and take the dual again.

**LP**

maximize:  $r$

subject to:  $\sum_{P \in \mathcal{P}_L(a, b, t)} R_{a, b, t}(P) = r$   
 $\forall a, b \in [N], t \in [T]$

$$\xi_{a, e} + \eta_{b, e} \geq \sum_{t=0}^{T-1} \sum_{P \in \mathcal{P}_L(a, b, t): e \in P} R_{a, b, t}(P)$$

$$\forall a, b \in [N], e \in E_{\text{phys}}$$

$$\sum_{a \in [N]} \xi_{a, e} + \sum_{b \in [N]} \eta_{b, e} \leq 1 \quad \forall e \in E_{\text{phys}}$$

$$\xi_{a, e} \geq 0 \quad \forall a \in [N], e \in E_{\text{phys}}$$

$$\eta_{b, e} \geq 0 \quad \forall b \in [N], e \in E_{\text{phys}}$$

$$R_{a, b, t}(P) \geq 0$$

$$\forall a, b \in [N], t \in [T], P \in \mathcal{P}_L(a, b, t)$$

### Dual

minimize:  $\sum_e z_e$

subject to:  $\sum_{a, b, t} x_{a, b, t} \geq 1$

$$z_e \geq \sum_b y_{a, b, e} \quad \forall a \in [N], e \in E_{\text{phys}}$$

$$z_e \geq \sum_a y_{a, b, e} \quad \forall b \in [N], e \in E_{\text{phys}}$$

$$\sum_{e \in P} y_{a, b, e} \geq x_{a, b, t}$$

$$\forall a, b \in [N], t \in [T], P \in \mathcal{P}_L(a, b, t)$$

$$y_{a, b, e}, z_e \geq 0 \quad \forall a, b \in [N], e \in E_{\text{phys}}$$

The variables  $y_{a, b, e}$  can be interpreted as either edge costs we assign dependent on source-destination pairs  $(a, b)$ , or demand functions designed to overload a particular edge  $e$ . We will use both interpretations, depending on if we are comparing  $y_{a, b, e}$  variables to either  $x_{a, b, t}$  or  $z_e$  variables respectively. According to the fourth dual constraint, the variables  $x_{a, b, t}$  can be interpreted as encoding the minimum cost of a path from  $(a, t)$  to  $\llbracket b \rrbracket$  subject to latency bound  $L$ . According to the second and third dual constraints, the variables  $z_e$  can be interpreted as bounding the throughput requested by the demand function  $D(t, a, b) = y_{a, b, e}$ . We will next define the cost inflation scheme we use to set our dual variables.

**Cost inflation scheme** For a given node  $a \in [N]$  and cutoff  $\theta \in \mathbb{Z}_{>0}$ , we will classify edges  $e$  according to whether they are reachable within  $\theta$  physical hops of  $a$ , counting edge  $e$  as one of the hops. (In other words, one could start at node  $a$  and cross edge  $e$  using  $\theta$  or fewer physical hops.) We define this value  $m_{\theta}^+(e, a)$  as follows.

$$m_{\theta}^+(e, a) = \begin{cases} 1 & \text{if } e \text{ can be reached from } a \text{ using at most } \theta \\ & \text{physical hops (including } e) \\ 0 & \text{if otherwise} \end{cases}$$

We define a similar value for edges which can reach node  $b$ .

$$m_{\theta}^-(e, b) = \begin{cases} 1 & \text{if } b \text{ can be reached from } e \text{ using at most } \theta \\ & \text{physical hops (including } e) \\ 0 & \text{if otherwise} \end{cases}$$

To understand how these values are set, consider some path  $P$  from  $(a, t) \rightarrow \llbracket b \rrbracket$ . If we consider the  $m_{\theta}^+, m_{\theta}^-$  weights on the edges of  $P$ , then the first  $\theta$  physical hop edges of  $P$  have weight  $m_{\theta}^+(e, a) = 1$  and the last  $\theta$  physical hop edges of  $P$  have weight  $m_{\theta}^-(e, b) = 1$ . It may be the case that some edges have both  $m_{\theta}^+(e, a) = m_{\theta}^-(e, b) = 1$ , if  $P$  uses fewer than  $2\theta$  physical hops. And if  $P$  uses  $\theta$  or fewer physical hops, then every physical hop edge along  $P$  has weight  $m_{\theta}^+(e, a) = m_{\theta}^-(e, b) = 1$ . All other weights may be 0 or 1 depending on whether those edges are otherwise reachable from  $a$  or can otherwise reach  $b$ .

We start by setting  $\hat{y}_{a, b, e} = m_{\theta}^+(e, a) + m_{\theta}^-(e, b)$ . Also set  $\hat{x}_{a, b, t} = \min_{P \in \mathcal{P}_L(a, b, t)} \{ \sum_{e \in P} \hat{y}_{a, b, e} \}$ . Note that by definition,  $\hat{x}$  and  $\hat{y}$  variables satisfy the last dual constraint. We will next find a lower

bound  $w \leq \sum_{a,b,t} \hat{x}_{a,b,t}$  and use that to normalize the  $\hat{x}, \hat{y}$  variables to satisfy the first dual constraint.

Note that  $\sum_{e \in P} \hat{y}_{a,b,e} \geq \min\{2\theta, 2|P \cap E_{\text{phys}}|\}$ . Then we can bound the sum of  $\hat{x}$  variables by

$$\sum_{a,b,t} \hat{x}_{a,b,t} \geq \sum_{a,t} \sum_{b \neq a} \min_{P \in \mathcal{P}_L(a,b,t)} \{2\theta, 2|P \cap E_{\text{phys}}|\}$$

Note that  $\hat{x}_{a,b,t} < 2\theta$  only when there exists some path from  $(a, t)$  to  $[[b]]$  which uses less than  $\theta$  physical edges. We can then use the Counting Lemma to produce an upper bound on the number of  $b \neq a$  which have such paths: this is at most  $2\binom{L}{\theta-1}$ .

So, assuming that  $2\binom{L}{\theta-1} \leq N$  and that  $\theta - 1 \leq L/3$ , we have

$$\sum_{a,t} \sum_{b \neq a} \hat{x}_{a,b,t} \geq NT \left( 2\theta \left( N - 2\binom{L}{\theta-1} \right) + \binom{L}{\theta-1} \right)$$

Set

$$w = NT \left( 2\theta \left( N - 2\binom{L}{\theta-1} \right) + \binom{L}{\theta-1} \right),$$

and then set  $y_{a,b,e} = \frac{1}{w} \hat{y}_{a,b,e}$  and  $x_{a,b,t} = \frac{1}{w} \hat{x}_{a,b,t}$ .

Next, we set  $z_e = \max_{a,b} \{\sum_a y_{a,b,e}, \sum_b y_{a,b,e}\}$ . By construction, the values of  $x_{a,b,t}, y_{a,b,e}, z_e$  that we have defined satisfy the dual constraints. Then to bound throughput from above, we upper bound the sums  $\sum_a y_{a,b,e}$  and  $\sum_b y_{a,b,e}$ , thus upper bounding the sum of  $z_e$ 's.

$$\sum_a y_{a,b,e} \leq \frac{1}{w} \left( \sum_a m_{\theta}^+(e, a) + N - 1 \right) \leq \frac{1}{w} \left( 2\binom{L}{\theta-1} + N - 1 \right)$$

where the last step is an application of the Counting Lemma. Similarly,

$$\sum_b y_{a,b,e} \leq \frac{1}{w} \left( N - 1 + 2\binom{L}{\theta-1} \right)$$

Recalling that  $z_e = \max_{a,b} \{\sum_a y_{a,b,e}, \sum_b y_{a,b,e}\}$ , we deduce that

$$z_e \leq \frac{1}{w} \left( N - 1 + 2\binom{L}{\theta-1} \right).$$

Using this upper bound on  $z_e$ , we find that the optimal value of the dual objective – hence also the optimal value of the primal, i.e. the maximum throughput of oblivious routing schemes – is bounded by

$$\begin{aligned} r &\leq \sum_e z_e \leq \frac{NT}{w} \left( N - 1 + 2\binom{L}{\theta-1} \right) \\ &\leq \frac{1}{2\theta} + \frac{4L^{\theta-1}}{2\theta(N(\theta-1)! - 2L^{\theta-1})} \end{aligned}$$

using the fact that  $\frac{a!}{(a-b)!} \leq a^b$ . At this point, we can rearrange the inequality to isolate  $L$ .

$$\begin{aligned} r - \frac{1}{2\theta} &\leq \frac{4L^{\theta-1}}{2\theta(N(\theta-1)! - 2L^{\theta-1})} \\ \left( \frac{(r - \frac{1}{2\theta})2\theta N(\theta-1)!}{4 + (r - \frac{1}{2\theta})4\theta} \right)^{\frac{1}{\theta-1}} &\leq L \end{aligned}$$

Now that we have a closed form, we simplify. We use Stirling's approximation, in the form  $(k!)^{\frac{1}{k}} \geq \frac{k}{e} \sqrt{2\pi k}^{\frac{1}{k}}$ .

$$\begin{aligned} L &\geq N^{\frac{1}{\theta-1}} (\theta-1)!^{\frac{1}{\theta-1}} \left( \frac{(r - \frac{1}{2\theta})2\theta}{4 + (r - \frac{1}{2\theta})4\theta} \right)^{\frac{1}{\theta-1}} \\ &\geq \frac{\theta-1}{e} N^{\frac{1}{\theta-1}} \left( \frac{(r - \frac{1}{2\theta})\theta \sqrt{\frac{\pi(\theta-1)}{2}}}{\theta r + \frac{1}{2}} \right)^{\frac{1}{\theta-1}} \end{aligned}$$

To set the parameter  $\theta$ , first note that the above bound is positive when  $r > \frac{1}{2\theta}$ . Additionally, we would like to set  $\theta$  as large as possible, and  $\theta$  must be an integer value (otherwise the Counting Lemma doesn't make sense). Taking this into account, we set  $\theta = \lfloor \frac{1}{2r} \rfloor + 1$ , the nearest integer for which  $(r - \frac{1}{2\theta})$  produces a positive value.

To simplify our lower bound further, let  $h = \lfloor \frac{1}{2r} \rfloor$  and  $\varepsilon = h + 1 - \frac{1}{2r}$ . These can be interpreted in the following way:  $h$  represents the largest number of physical hops we take per path (approximately), and  $\varepsilon$  is directly related to how many pairs take paths using  $h$  physical hops instead of paths using fewer than  $h$  physical hops. Note that  $\varepsilon \in (0, 1]$ . This gives the restated bound below.

$$\begin{aligned} L &\geq \frac{h}{e} N^{1/h} \left( \frac{(r - \frac{1}{2(h+1)})(h+1)\sqrt{\pi h/2}}{(h+1)r + \frac{1}{2}} \right)^{1/h} \\ &\geq \frac{h}{e} (\varepsilon N)^{1/h} \left( \frac{\sqrt{\pi h/2}}{4h} \right)^{1/h} \\ &= \Omega \left( h(\varepsilon N)^{1/h} \right) \end{aligned} \tag{2}$$

As  $\varepsilon \rightarrow 0$ , this bound goes toward 0, making it meaningless for extremely small values of  $\varepsilon$ . However, for such values of  $\varepsilon$ , we simply set  $\theta = h + 2$  instead, which gives the following

$$L_{\max} \geq \Omega \left( (h+1)N^{1/(h+1)} \right)$$

To combine the two ways in which we set  $\theta$ , we take the average of the two bounds. This gives the bound from our theorem statement,

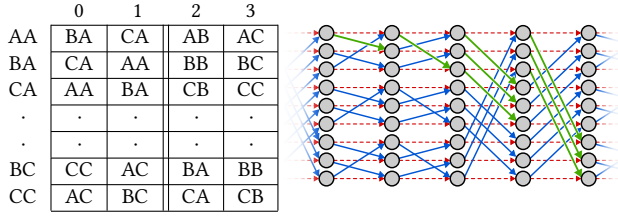
$$L_{\max} \geq \Omega \left( h \left[ (\varepsilon N)^{1/h} + N^{1/(h+1)} \right] \right) = \Omega \left( L^*(r, N) \right).$$

□

## 4 UPPER BOUND

To prove an upper bound on the latency achievable while guaranteeing a given throughput, we define an infinite family of ORN designs which we refer to as the Elementary Basis Scheme (EBS). The upper bound given by EBS is within a constant factor of  $L^*(r, N)$  for most values of  $r$ . To tightly bound the remaining values of  $r$ , we describe a second infinite family of ORN designs which we refer to as the Vandermonde Bases Scheme (VBS). Combined, EBS and VBS give a tight upper bound on maximum latency for all constant  $r$ . In the full version of this paper [1], we address the upper bound for  $d$ -regular networks with  $d > 1$  by modifying EBS and VBS.





**Figure 3: Connection schedule for 9 nodes in  $h = 2$  EBS, as well as part of the corresponding virtual topology. Physical edges used on semi-paths from  $((AA),0)$  to other nodes are highlighted in green. This schedule can be seen as a generalization of the one presented in Figure 2.**

#### 4.1 Elementary Basis Scheme

*Connection Schedule:* Each node will participate in a series of sub-schedules called round robins. Consider the cyclic group  $H = \mathbb{Z}/(n)$  acting freely on a set  $S$  of  $n$  nodes, where we denote the action of  $t \in H$  on  $i \in S$  by  $i + t$ . A round robin for  $S$  is a schedule of  $n - 1$  timeslots in which each element of  $S$  has a chance to send directly to each other element exactly once; during timeslot  $t \in [n - 1]$ , node  $i$  may send to  $i + t$ . The number of round-robins in which each EBS node participates is controlled by a tuning parameter  $h$  which we refer to as the *order*. Similar to in Section 3,  $h$  will be half of the the maximum number of physical hops in an EBS path.

Let  $n = N^{1/h}$ , so that the node set  $[N]$  is in one-to-one correspondence with the elements of the group  $H^h$ . Each node  $a \in [N]$  is assigned a unique set of  $h$  coordinates  $(a_0, a_1, \dots, a_{h-1}) \in H^h$  and participates in  $h$  round robins, each containing the  $n$  nodes that match in all but one of the  $h$  coordinates. We refer to these round robins as *phases* of the EBS schedule. One full iteration of the EBS schedule, or an *epoch*, contains  $h$  phases. Because each phase is a round robin among  $n$  nodes, each phase takes  $n - 1$  timeslots, resulting in an overall epoch length of  $T = h(n - 1) = h(N^{1/h} - 1)$ .

We now describe the EBS schedule formally. We express each node  $i$  as the  $h$ -tuple  $(i_0, i_1, \dots, i_{h-1}) \in (\mathbb{Z}/n)^h$ . Similarly, we identify each permutation  $\pi_k$  of the connection schedule using a scale factor  $s$ ,  $1 \leq s < n$ , and a phase number  $p$ ,  $0 \leq p < h$ , such that  $k = (n - 1)p + s - 1$ . Let  $\mathbf{e}_p$  denote the standard basis vector whose  $p^{\text{th}}$  coordinate is 1 and all other coordinates are 0. The connection schedule is then  $\pi_{(n-1)p+s-1}(i) = i + s\mathbf{e}_p = j$ . Since  $\mathbf{e}$  is the standard basis,  $j_x = i_x$  for  $x \neq p$ , and  $j_p = i_p + s \pmod{n}$ .

The EBS schedule can be seen as simulating a flattened butterfly graph between nodes [15]. This schedule generalizes existing ORN designs which have thus far all been based on the same schedule: a single round robin among all nodes, simulating an all-to-all graph. When  $h = 1$ , the EBS schedule reduces to this existing schedule. On the other hand, when  $h = \log_2(N)$ , the EBS schedule simulates a direct-connect hypercube topology. By varying  $h$ , in addition to achieving these two known points, the EBS family includes schedules which achieve intermediate throughput and latency tradeoff points.

**4.1.1 Oblivious Routing Scheme.** The EBS oblivious routing scheme is based around Valiant load balancing (VLB) [26]. VLB operates in two stages: first, traffic is routed from the source to a random

intermediate node in the network. Then, traffic is routed from the intermediate node to its final destination. This two-stage design ensures that traffic is uniformly distributed throughout the network regardless of demand. We refer to the path taken during an individual stage as a *semi-path*, and we use the same algorithm to generate semi-paths in either stage.

To create a semi-path between a node  $(a, t)$  and  $\llbracket b \rrbracket$ , the following greedy algorithm is used starting at  $(a, t)$ : for the current node in the virtual topology, if the outgoing physical edge leads to a node with a decreased Hamming distance to  $b$  (i.e. it matches  $b$  in the modified coordinate), traverse the physical edge. Otherwise, traverse the virtual edge. This algorithm terminates when it reaches a node in  $\llbracket b \rrbracket$ . Note that because there are  $h$  coordinates, the largest Hamming distance possible is  $h$ , and the longest semi-paths use  $h$  physical links.

In order to construct a full path from  $(a, t)$  to  $\llbracket b \rrbracket$ , first select an intermediate node  $c$  in the system uniformly at random. Then, traverse the semi-path from  $(a, t)$  to  $\llbracket c \rrbracket$ . Let  $t'$  be the timeslot at which we reach  $\llbracket c \rrbracket$ . If  $t' < t + T$ , traverse virtual edges until node  $(c, t + T)$  is reached. Finally, traverse the semi-path from  $(c, t + T)$  to  $\llbracket b \rrbracket$ .

The EBS oblivious routing scheme is formed as follows: for  $R_{a,b,t}$ , for all intermediate nodes  $c$ , construct the path from  $(a, t)$  to  $\llbracket b \rrbracket$  via  $c$  as described above, and assign it the value  $\frac{1}{N}$ . Assign all other paths the value 0. Because there are  $N$  possible intermediate nodes, each of which is used to define one path from  $(a, t)$  to  $\llbracket b \rrbracket$ , this routing scheme defines one unit of flow.

#### 4.2 Latency-Throughput Tradeoff of EBS

**Proposition 1.** *For each  $r \leq \frac{1}{2}$  such that  $h = \frac{1}{2r}$  is an integer, and each  $N > 1$  such that  $N^{1/h}$  is an integer, the EBS design of order  $h$  on  $N$  nodes guarantees throughput  $r$  and has maximum latency  $\frac{1}{r}(N^{2r} - 1)$ .*

The proof of Proposition 1 is contained in the following two subsections, which address the latency and throughput guarantees respectively.

**4.2.1 Latency.** Recall that  $h = \frac{1}{2r}$  and that  $n = N^{1/h} = N^{2r}$ , so the latency bound in Proposition 1 can be written as  $2h(n - 1)$ . Since the epoch length is  $T = h(n - 1)$ , the latency bound asserts that every EBS routing path completes within a time interval no greater than the length of two epochs. An EBS path is composed of two semi-paths, so we only need to show that each semi-path completes within the length of a single epoch.

Let  $(a, t)$  denote the first node of the semi-path. If  $t$  occurs at the start of a phase, then after  $p$  phases have completed the Hamming distance to the semi-path's destination address must be less than or equal to  $t - p$ ; consequently the semi-path completes after at most  $h$  phases, as claimed. If  $t$  occurs in the middle of a phase using basis vector  $\mathbf{e}_p$ , let  $s$  denote the number of timeslots that have already elapsed in that phase. Either the semi-path is able to match the  $p^{\text{th}}$  destination coordinate before the phase ends, or the coordinate can be matched during the first  $s$  timeslots of the next phase that uses basis vector  $\mathbf{e}_p$ . In either case, the  $p^{\text{th}}$  destination coordinate will be matched no later than timeslot  $t + T$ , and all other destination coordinates will be matched during the intervening phases.

### 4.2.2 Throughput.

**Lemma 2.** *Let  $R$  be the EBS routing scheme for a given  $N$  and  $h$ . For all demand functions  $D$  requesting throughput at most  $\frac{1}{2h}$ , the flow  $f(R, D)$  is feasible.*

**PROOF.** Consider an arbitrary demand function  $D$  requesting throughput  $r = \frac{1}{2h}$ , and consider an arbitrary physical edge  $e \in E_{\text{phys}}$  from  $(i, t_e)$  to  $(j, t_e + 1)$ , where  $t_e$  is the timeslot during which the edge begins. Let  $t_e \equiv (p_e, s_e)$  such that  $p_e$  is the phase in the schedule corresponding to  $t_e$ , and  $s_e$  is the scale factor used during  $t_e$ . We wish to show that  $F(f(R, D), e) \leq 1$ .

We first use a greedy algorithm (described in the full version of this paper [1]) to generate  $D'$ , a demand function such that for all  $t$ ,  $D'(t)$  has row and column sums exactly equal to  $r$ , and  $D'(t)$  bounds  $D(t)$  above. Due to the latter condition, it follows that  $f(R, D')$  bounds  $f(R, D)$  above; thus  $F(f(R, D'), e) \geq F(f(R, D), e)$ . Henceforward, we focus on proving  $F(f(R, D'), e) \leq 1$ .

Valid paths in EBS include two components: the semi-path from the source node to an intermediate node, and the semi-path from the intermediate node to the destination node. We can therefore decompose the paths in  $F(f(R, D'), e)$  into two components as follows: first, we define  $R'$ , a routing protocol defined such that  $R'_{a,b,t}(P)$  equals 1 if  $P$  is the semi-path from  $(a, t)$  to  $\llbracket b \rrbracket$ , and 0 otherwise. Because EBS uses the same routing strategy for both source-intermediate semi-paths and intermediate-destination semi-paths,  $R'$  is used for both components. Then, we introduce two demand functions:  $D'_{a \rightarrow b}$  represents demand on semi-paths from origin nodes to intermediate nodes, while  $D'_{b \rightarrow c}$  represents demand on semi-paths from intermediate nodes to destination nodes. Note that for all physical edges  $e$ ,

$$F(f(R, D'), e) = F(f(R', D'_{a \rightarrow b}), e) + F(f(R', D'_{b \rightarrow c}), e).$$

To characterize  $D'_{a \rightarrow b}$ , note that regardless of source and destination,  $R$  samples intermediate nodes uniformly. Therefore, for all  $(t, a, b) \in \mathbb{Z} \times [N] \times [N]$ ,

$$D'_{a \rightarrow b}(t, a, b) = \frac{1}{N} \sum_{c \in [N]} D'(t, a, c) = \frac{r}{N}$$

Similarly, because semi-paths from an intermediate node to the destination always commence exactly  $T$  timeslots after the starting vertex, we can characterize  $D'_{b \rightarrow c}(t, b, c)$  as follows:

$$D'_{b \rightarrow c}(t, b, c) = \frac{1}{N} \sum_{a \in [N]} D'(t - T, a, c) = \frac{r}{N}$$

Note that  $D'_{a \rightarrow b} = D'_{b \rightarrow c} = D^{ALL}$ , where  $D^{ALL}$  is the uniform all-to-all demand function  $D^{ALL}(t, a, b) = \frac{r}{N}$  for all  $(t, a, b) \in \mathbb{Z} \times [N] \times [N]$ . Therefore,  $F(f(R, D), e) \leq 2F(f(R', D^{ALL}), e)$ .

**Claim 1.** *For all  $e \in E_{\text{phys}}$ , there are exactly  $Tn^{h-1}$  triples  $(t, a, b)$  such that the semi-path from  $(a, t)$  to  $\llbracket b \rrbracket$  traverses  $e$ .*

**PROOF OF CLAIM.** Denote the endpoints of edge  $e$  by  $(i, t_e)$  and  $(i + s \cdot e_p, t_e + 1)$ . The semi-path of a triple  $(t, a, b)$  traverses  $e$  if and only if the semi-path first routes from  $(a, t)$  to  $(i, t_e)$ , and  $(b - a)_p = s$ .

Because semi-paths complete in  $T$  timeslots, only semi-paths beginning in timeslots in the range  $[t_e - T + 1 \dots t_e]$  could possibly

reach node  $(i, t_e)$  and traverse  $e$ . For every  $t \in [t_e - T + 1 \dots t_e]$ , where  $t \equiv (p_t, s_t)$ , we can construct  $n^{h-1}$  such triples as follows: First, we select  $\mathbf{d}$ , a vector representing the difference between  $a$  and  $b$  in the triple we will construct. To satisfy the second condition on  $(t, a, b)$ , we must set  $\mathbf{d}_p = s$ . However, the remaining  $h - 1$  indices of  $\mathbf{d}$  can take on any of the  $n$  possible values. Thus, there are  $n^{h-1}$  possibilities for  $\mathbf{d}$ .

For any semi-path  $(t, a, b)$  such that  $b - a = \mathbf{d}$ , the timeslots in which a physical edge is traversed can be determined from  $\mathbf{d}$ . For any given timeslot  $t' \equiv (p', s')$  such that  $t \leq t' < t + T$ , a physical edge is traversed if and only if  $\mathbf{d}_{p'} = s'$ . These are the edges that decrease the Hamming distance to  $b$  by correctly setting coordinate  $p$ . We thus construct  $a$  as follows: For every index  $p$ , if  $(\mathbf{d}_p, p)$  is between  $k_t$  and  $k_e - 1$  inclusive, we set  $a_p = i_p - \mathbf{d}_p$ . Otherwise, we set  $a_p = i_p$ . Once we have constructed  $a$ ,  $b$  is simply  $a + \mathbf{d}$ . This choice of  $a$  and  $b$  ensures that by timeslot  $t_e$ , the semi-path from  $(a, t)$  to  $\llbracket b \rrbracket$  reaches  $\llbracket i \rrbracket$ .

For each of the  $T$  timeslots for which semi-paths originating in the given timeslot may traverse  $e$ , there are  $n^{h-1}$  such semi-paths. This gives a total of  $Tn^{h-1}$  semi-paths that traverse  $e$  over all timeslots. Note that because each such semi-path has a unique  $(t, \mathbf{d})$ , none of the constructed semi-paths are double counted. In addition, because the  $(t, \mathbf{d})$  pair determines the timeslots in which physical links are followed, and because there is only one physical link entering and leaving each node during each timeslot, there cannot be more than one choice of  $a$  for a given  $(t, \mathbf{d})$  pair such that the semi-path includes  $(i, t_e)$ . Because the  $Tn^{h-1}$  count includes all possible choices of  $\mathbf{d}$  for every timeslot, all semi-paths that traverse  $e$  are accounted for.  $\square$

Now we continue with the proof of Lemma 2. Since exactly  $Tn^{h-1}$  triples  $(t, a, b)$  correspond to semi-paths that traverse  $e$ , and  $D^{ALL}$  assigns  $\frac{r}{N}$  flow to each semi-path,  $F(f(R', D^{ALL}), e) = \frac{r}{N} Tn^{h-1} = \frac{r}{N} h(n-1)n^{h-1}$ . Thus:

$$F(f(R, D), e) \leq 2F(f(R', D^{ALL}), e) = 2 \frac{r}{N} h(n-1)n^{h-1} < 2rh$$

When  $r \leq \frac{1}{2h}$ , for all physical edges  $e$ ,  $F(f(R, D), e) \leq 1$ . Thus,  $f(R, D)$  is feasible.  $\square$

### 4.2.3 Tightness of EBS Upper Bound.

**Lemma 3.** *For  $0 < r \leq \frac{1}{2}$  let  $h = \lfloor \frac{1}{2r} \rfloor$  and  $\varepsilon = h + 1 - \frac{1}{2r}$ . The EBS design of order  $h$  attains maximum latency at most  $CL^*(r, N)$ , except when*

$$\varepsilon \geq 2\sqrt{\frac{2h}{\pi}} \left(\frac{2e}{C}\right)^h.$$

**PROOF.** Theorem 2 and Proposition 1 together show the following about the maximum latency of EBS compared to the maximum latency lower bound:

$$L_{\text{EBS}} \leq 2hN^{1/h}$$

$$L^*(r, N) \geq \frac{h}{e} (\varepsilon N)^{1/h} \left(\frac{\sqrt{\pi h/2}}{4h}\right)^{1/h}$$

Note that this interpretation of the maximum latency lower bound is taken from equation (2) in the proof of Theorem 2.

Suppose we wish to assert  $L_{EBS}/L^*(r, N) \leq C$ . Given  $C$  and  $h$ , we will derive the possible values of  $\varepsilon$  for which this assertion holds.

$$C \geq \frac{2hN^{1/h}}{\frac{h}{e}(\varepsilon N)^{1/h} \left(\frac{\sqrt{\pi h/2}}{4h}\right)^{1/h}} = \frac{2e}{\left(\frac{\varepsilon\sqrt{\pi h/2}}{4h}\right)^{1/h}}$$

$$\iff \varepsilon \geq 2\sqrt{\frac{2h}{\pi}} \left(\frac{2e}{C}\right)^h$$

□

When  $\varepsilon$  falls outside this range, the maximum latency of the EBS design is far from optimal. In the following sections we present and analyze an ORN design which gives a tighter upper bound when  $\varepsilon$  falls outside this range, in other words when  $\varepsilon < 2\sqrt{\frac{2h}{\pi}} \left(\frac{2e}{C}\right)^h$ .

### 4.3 Vandermonde Bases Scheme

In order to provide a tight bound when  $\varepsilon$  is very small, we define a new family of ORN designs which we term the Vandermonde Bases Scheme (VBS). VBS is defined for values of  $N$  which are perfect powers of prime numbers. We begin by providing some intuition behind the design of VBS.

For  $h = \lfloor \frac{1}{2r} \rfloor$  and  $\varepsilon = h + 1 - \frac{1}{2r}$ , a small value of  $\varepsilon$  indicates that  $r$  is slightly above  $\frac{1}{2(h+1)}$ . This indicates that the average number of physical hops in a path can be at most slightly below the even integer  $2(h+1)$ . EBS is only able to achieve an average number of physical hops equal to an even integer as  $N$  becomes sufficiently large. In small  $\varepsilon$  regions, the difference between the highest average number of physical hops theoretically capable of guaranteeing  $r$  throughput and the average number of physical hops used by EBS approaches 2. This suggests that EBS achieves a throughput-latency tradeoff that favors throughput more than is necessary in these regions, penalizing latency too much to form a tight bound. A more effective ORN design for these regions would usually use paths with  $2(h+1)$  physical hops, and mix in sufficiently many paths with fewer physical hops to ensure that the average number of physical hops per path is at most  $2(h+1-\varepsilon)$ .

VBS achieves this by employing two routing strategies for semi-paths alongside each other. The first strategy, single-basis (SB) paths, resembles the semi-path routing used by EBS for  $h' = h + 1$ . The second strategy, hop-efficient (HE) paths, will rely on the fact that VBS's schedule regularly modifies the basis used to determine which nodes are connected to one another. HE paths will consider edges beyond the current basis, enabling them to form semi-paths between nodes using only  $h$  hops, even when this is not possible within a single basis. The more future phases are considered, the more nodes can be connected by HE paths. This tuning provides a high granularity in the achieved tradeoff between throughput and latency, and enables a tight bound in regions where  $\varepsilon$  is small. It is interesting that the quantitative reasoning underlying this scheme is reminiscent of the proof of the Counting Lemma (Lemma 1), which similarly classifies paths into short paths and long paths and counts the number of destinations reachable by short paths.

We define VBS for  $N = n^{h+1}$  such that  $n$  is a prime number. The connection schedule and routing algorithm of VBS depend on a parameter  $\delta$ , which represents a target for the fraction of semi-paths

that traverse HE paths. We later describe how to set  $Q$ , the number of future phases considered for HE paths, such that the number of destinations reachable by HE paths is approximately  $\delta N$ .

**4.3.1 Connection Schedule.** We assume the total number of nodes in the system is  $N = n^{h+1}$  for some prime number  $n$ . As in EBS, each node  $a$  is assigned a unique set of  $h+1$  coordinates  $(a_0, a_1, \dots, a_h)$ , each ranging from 0 to  $n-1$ . This maps each node to a unique element of  $\mathbb{F}_n^{h+1}$ . We identify each permutation  $\pi_k$  of the connection schedule using a scale factor  $s$ ,  $1 \leq s < n$  and a phase number<sup>3</sup>  $p$ ,  $0 \leq p < n$ , such that  $k = (n-1)p + s - 1$ . Each phase  $p$  is formed using the Vandermonde vector  $\mathbf{v}(p) = (1, p, p^2, \dots, p^h)$ . This produces the connection schedule  $\pi_{(n-1)p+s-1}(\mathbf{i}) = \mathbf{i} + s\mathbf{v}(p)$ .

**4.3.2 Routing Algorithm.** As with EBS, VBS's oblivious routing scheme is based around VLB. First, traffic is routed along a semi-path from the source to a random intermediate node in the network, and then traffic is routed along a second semi-path from the intermediate node to its final destination. As in EBS, the same algorithm is used to generate semi-paths in both stages of VLB. However, unlike in EBS, semi-paths are only defined starting at phase boundaries. Thus, the first step of a VBS path is to traverse up to  $n-2$  virtual edges until a phase boundary is reached. Paths are then defined for a given  $(q, a, b)$  triple, where  $q = t/(n-1)$  for some timeslot  $t$  at the beginning of a phase (hence  $t$  is divisible by  $n-1$ ). Following the initial virtual edges to reach a phase boundary, we concatenate the semi-path from the source to the intermediate node, followed by the semi-path from the intermediate node to the destination.

Depending on the current phase and the source-destination pair, we either route via a single-basis (SB) path or a hop-efficient (HE) path. The routing scheme always selects a HE semi-path when one is available, and otherwise it selects a SB path. We describe both path types below.

**Single-basis paths** - The SB path, for a given  $(q, a, b)$  is formed as follows: First, we define the distance vector  $\mathbf{d} = b - a$ , as well as the basis  $Y = (v(q), v(q+1), \dots, v(q+h))$ . Note that the vectors in the basis  $Y$  are those used to form the  $h+1$  phases beginning with phase  $q$ . Then, we find  $\mathbf{s} = Y^{-1}\mathbf{d}$ . Over the next  $h+1$  phases, for every timeslot  $t' \equiv (p', s')$ , if  $s' = s_p$ , the physical edge is traversed. Otherwise, the virtual edge is traversed. This strategy corresponds to traversing  $\mathbf{d}$  through its decomposition in basis  $Y$ , beginning at node  $a$  and ending at node  $b$ .

This algorithm for SB paths completes within  $h+1$  phases. However, to ensure that both SB and HE paths take  $h+1+Q$  phases to complete, following this virtual edges are traversed for a further  $Q$  phases. Note that an SB path may have fewer than  $h+1$  hops, although this becomes increasingly rare as  $N$  grows without bound.

**Hop-efficient paths** - A HE path, is formed as follows: First, for  $h+1$  phases, only virtual edges are traversed. This ensures that the physical hops of HE and SB paths beginning during the same phase  $q$  use disjoint sets of vectors (assuming  $n > h+1+Q$ ), which simplifies later analysis. Following this initial buffer period,  $h$  phases are selected out of the next  $Q$  phases, and one physical hop is taken in each selected phase. During all other timeslots within the  $Q$  phases, virtual hops are taken.

<sup>3</sup>The mnemonic is that  $p$  stands for "phase number", not "prime number". We beg the forgiveness of readers who find it confusing that the size of the prime field is denoted by  $n$ , not  $p$ .

For a given starting phase  $q$  and node  $a$ , there are  $\binom{Q}{h}(n-1)^h$  possible HE paths. Recall that we would like  $\delta N$  destinations to be reachable by HE paths. Ignoring for now the possibility of destinations reachable by multiple HE paths, we set  $Q$  to the lowest integer value such that:

$$\binom{Q}{h}(n-1)^h \geq \delta N \iff \binom{Q}{h} \geq \delta n$$

Note that for this value of  $Q$ ,  $\binom{Q-1}{h} < \delta n$ . For some  $(q, a, b)$ , more than one HE path may exist. In this case, one of these paths is arbitrarily selected; this selection does not affect our analysis.

#### 4.4 Latency-Throughput Tradeoff of VBS

**4.4.1 Latency.** A VBS path begins with at most  $n-2$  virtual edges traversed until a phase boundary is reached. Following this, the first semi-path immediately begins, followed by the second semi-path. Because both SB and HE paths are defined to take  $h+1+Q$  phases, the total maximum latency of VBS paths is  $(n-2)+2(n-1)(h+1+Q) = (n-1)(3+2h+2Q) - 1$ .

**4.4.2 Throughput.**

**Lemma 4.** *Let  $R$  be the VBS routing scheme for a given  $N, h$ , and  $\delta$ , such that  $\delta \leq \frac{1}{4(h+1)(1+\frac{1}{2h})^2}$ . For all demand functions  $D$  requesting throughput at most  $\frac{1}{2(h+1-\epsilon)}$  for  $\epsilon = \frac{1}{4}\delta$ , the flow  $f(R, D)$  is feasible.*

**PROOF.** Consider an arbitrary demand function  $D$  requesting throughput at most  $r$ , and consider an arbitrary physical edge  $e \in W_{\text{phys}}$  from  $(i, t_e)$  to  $(j, t_e + 1)$ , where  $t_e$  is the timeslot during which the edge begins. Let  $t_e \equiv (p_e, s_e)$  such that  $p_e$  is the phase in the schedule corresponding to  $t_e$ , and  $s_e$  is the scale factor used during  $t_e$ . We wish to show that  $F(f(R, D), e) \leq 1$ .

As in our proof of Lemma 2, we begin by inflating  $D$  into  $D'$ . Similarly, we define  $R'$ , the routing protocol for semi-paths, and we decompose  $f(R, D')$  into  $f(R', D'_{a \rightarrow b})$  and  $f(R', D'_{b \rightarrow c})$ . Note that because semi-paths begin only on phase boundaries,  $R'$  does not strictly follow our definition for an oblivious routing scheme. Instead, we define  $R'_{a,b,q}$  using phases  $q$ , rather than timeslots  $t$ , for the domain. The paths used for  $R'_{a,b,q}$  begin during the first timeslot of phase  $q$ . This is reflective of the definition of semi-paths in VBS.

To generate  $D'_{a \rightarrow b}$ , note that  $R$  first batches  $(a, b, t)$  triples over the  $n-1$  timeslots preceding an epoch boundary, before sampling intermediate nodes uniformly. Therefore, for all  $(q, a, b)$

$$D'_{a \rightarrow b}(q, a, b) = \frac{1}{N} \sum_{t \in [n-1]} \sum_{c \in [N]} D'(q(n-1) - t, a, c) = \frac{(n-1)r}{N}$$

Similarly, because semi-paths from an intermediate node to the destination always begin exactly  $h+1+Q$  phases after the beginning of the first semi-path, we can define  $D'_{b \rightarrow c}(t, b, c)$  as follows:

$$D'_{b \rightarrow c}(q, b, c) = \frac{1}{N} \sum_{t \in [n-1]} \sum_{c \in [N]} D'((q-h-1-Q)(n-1) - t, a, c) = \frac{(n-1)r}{N}$$

Note that  $D'_{a \rightarrow b} = D'_{b \rightarrow c} = D^{ALL}$ , where  $D^{ALL}$  is the uniform all-to-all demand function  $D^{ALL}(q, a, b) = \frac{(n-1)r}{N}$  for all  $(q, a, b) \in \mathbb{Z} \times [N] \times [N]$ . Therefore,  $F(f(R, D), e) \leq 2F(f(R', D^{ALL}), e)$ .

To calculate  $F(f(R', D^{ALL}), e)$ , we compute the number of  $(q, a, b)$  triples that traverse edge  $e$ . We calculate this number as follows: First, we calculate  $\#_{SB}$ , which represents the number of  $(q, a, b)$  triples that have an SB path that traverses edge  $e$ . Then, we calculate  $\#_{miss}$ , the number of such triples that have an HE path available (and thus do not traverse  $e$ ). Finally, we determine  $\#_{HE}$ , the number of triples that traverse  $e$  using an HE path. The total flow traversing edge  $e$  is then  $F(f(R', D^{ALL}), e) = \frac{(n-1)r}{N} (\#_{SB} - \#_{miss} + \#_{HE})$ .

To find  $\#_{SB}$ , we use reasoning similar to that used in Lemma 2. In order for a given  $(q, a, b)$  to have an SB path that traverses edge  $e$ , the SB path for  $(q, a, b)$  must reach node  $(i, t)$ , then traverse edge  $e$ . The only values of  $q$  for which this is possible are those in the range  $q_e - h \leq q \leq q_e$ . For each of these  $q$ , we can generate  $n^h$  distinct  $(q, a, b)$  triples that have SB paths that traverse edge  $e$  as follows. First, select an arbitrary  $\mathbf{s}$  such that  $s_{q_e - q} = s_e$ . Then, set  $a = \mathbf{i} - \sum_{q'=q}^{q_e-1} s_{q'-q} v(q')$ , and  $b = a + \sum_{q'=q}^{q+h} s_{q'-q} v(q')$ . In this case,  $\mathbf{s}$  corresponds to a distance vector between  $a$  and  $b$ , expressed in terms of the basis used for SB paths starting in phase  $q$ . Because of how  $a$  is set, it is clear that the SB path for  $(q, a, b)$  must traverse  $(i, t)$ . In addition, because  $s_{q_e - q} = s_e$ , the SB path will traverse edge  $e$  instead of another edge during the same phase.

For a given  $q$ , there are  $n^h$  possible values for  $\mathbf{s}$ , because all but one of its  $h+1$  elements can be set to any value in  $[n]$ . There are  $(h+1)$  possible values for  $q$ , giving a total of  $\#_{SB} = (h+1)n^h$ .

To find  $\#_{miss}$ , we compare the distance vectors of  $(q, a, b)$  triples that have SB paths which traverse  $e$  with those of  $(q, a, b)$  triples that have valid HE paths. Each vector found in the overlap between these two sets corresponds to one triple that contributes to  $\#_{miss}$ . To reason about the former set of vectors, we return to the construction of  $\mathbf{s}$  used to find  $\#_{SB}$ . For a given starting phase  $q$ , each  $\mathbf{s}$  such that  $s_{q_e - q} = s_e$  represents a distance vector that can traverse  $e$ , expressed in terms of the basis used for SB paths starting in phase  $q$ . We can construct this basis as  $Y = (v(q), v(q+1), \dots, v(q+h))$ . For each  $\mathbf{s}$ ,  $\mathbf{d} = Y\mathbf{s}$  is the same distance vector expressed using the elementary basis. The range of distance vectors  $\mathbf{d}$  reachable while traversing  $e$  forms  $D_e$ , an  $h$ -dimensional affine subspace of  $\mathbb{F}_n^{h+1}$  that is parallel to  $W_e$ , the linear subspace spanned by  $Y \setminus \{v(q_e)\}$ .

Next, we consider which triples have valid HE paths. For a given starting phase  $q$ , there are  $Q$  phases which are considered for forming HE paths. Let  $I$  be a set of  $h$  phase numbers chosen from these  $Q$  phases, and let  $V(I)$  be the linear subspace spanned by the vectors corresponding to the phase numbers in  $I$ . There are  $\binom{Q}{h}$  ways of choosing such a set  $I$ . For each possible choice,  $V(I)$  forms an  $h$ -dimensional linear subspace in  $\mathbb{F}_n^{h+1}$ , corresponding to the distance vectors reachable via HE paths using the chosen phases. (Note that  $V(I)$  must be  $h$ -dimensional because every  $h$  distinct Vandermonde vectors are linearly independent.) Because  $V(I)$  and  $W_e$  are spanned by distinct sets of  $h$  Vandermonde vectors, these linear subspaces are not equivalent, implying that  $V(I)$  and  $D_e$  are not parallel. Thus,  $V(I) \cap D_e$  is an affine subspace with dimension  $h-1$  and contains  $n^{h-1}$  distance vectors.

Some distance vectors lie in more than one such intersection. To avoid overcounting  $\#_{miss}$ , we must remove at least this many vectors from our count. Given two sets  $I$  and  $J$  of  $h$  chosen phase numbers,  $V(I)$  and  $V(J)$  form two different linear subspaces of

$\mathbb{F}_n^{h+1}$ . As linear subspaces, both  $I$  and  $J$  contain the zero vector, as does the  $(h-1)$ -dimensional  $I \cap J$ .  $D_e$  does not contain the zero vector, so  $D_e \cap I \cap J$  can only be  $(h-2)$ -dimensional, containing  $n^{h-2}$  distance vectors. There are fewer than  $\binom{Q}{h}^2$  ways of choosing two distinct sets  $I$  and  $J$ .

Thus, for a given starting  $q$ , there are fewer than  $\binom{Q}{h} n^{h-1} - \binom{Q}{h}^2 n^{h-2}$  distance vectors in the overlap between  $D_e$  and the union of all possible  $V(I)$ . Because there are  $h+1$  possibilities for the starting  $q$ , this gives the following lower bound for  $\#_{miss}$ :

$$\begin{aligned} \#_{miss} &> (h+1) \left( \binom{Q}{h} n^{h-1} - \binom{Q}{h}^2 n^{h-2} \right) \\ &> (h+1) \left( \delta n^h - \delta^2 n^h \left( \frac{Q}{Q-h} \right)^2 \right) \end{aligned}$$

To find  $\#_{HE}$ , note that a given  $(q, a, b)$  can only traverse edge  $e$  if  $q_e - h - Q \leq q < q_e - h$ , since  $q_e$  must be in the set of  $Q$  phases considered for HE paths for  $(q, a, b)$ . For a given  $q$ , we can construct an HE path by selecting  $h-1$  additional phases from the  $Q-1$  remaining phases, and then selecting one of the  $n-1$  edges within that phase to traverse. Some of these paths may lead to the same destination, causing an overcount, but it is fine to overcount  $\#_{HE}$ .

$$\#_{HE} \leq Q \binom{Q-1}{h-1} (n-1)^{h-1} < \delta h n^h \frac{Q}{Q-h}$$

Having found  $\#_{SB}$ ,  $\#_{miss}$ , and  $\#_{HE}$ , we can now bound  $F(f(R, D), e)$ :

$$\begin{aligned} F(f(R, D), e) &\leq 2 \frac{(n-1)r}{N} (\#_{SB} - \#_{miss} + \#_{HE}) \\ &< 2r(h+1) \left( 1 - \delta \left( 1 - \frac{h}{h+1} \frac{Q}{Q-h} \right) + \delta^2 \left( \frac{Q}{Q-h} \right)^2 \right) \end{aligned}$$

For  $Q \geq 2h^2 - h$ ,  $\frac{Q}{Q-h} \leq \frac{h+\frac{1}{2}}{h}$ . This gives:

$$\begin{aligned} F(f(R, D), e) &< 2r(h+1) \left( 1 - \delta \left( 1 - \frac{h}{h+1} \frac{h+\frac{1}{2}}{h} \right) + \delta^2 \left( \frac{h+\frac{1}{2}}{h} \right)^2 \right) \\ &\leq \frac{1}{h+1-\varepsilon} (h+1-\varepsilon) = 1 \end{aligned}$$

Note that because of how we set  $\varepsilon$  and restrict  $\delta$ ,  $\varepsilon \leq \frac{1}{2}\delta - (h+1)\delta^2(1+\frac{1}{2h})^2$ . Because the amount of flow traversing any physical edge  $e$  is less than 1, the flow  $f(R, D)$  is feasible.  $\square$

#### 4.5 Tightness of Upper Bound

**Theorem 3.** *For all  $r \in (0, 1/2]$ , there is a VBS design or an EBS design which guarantees throughput  $r$  and uses maximum latency*

$$L_{max} \leq O(L^*(r, N)). \quad (3)$$

**PROOF.** The VBS design of order  $h$  with parameter  $\delta$  gives maximum latency  $L \leq (h+1)(n-1) + Q(n-1)$  for  $h = \lfloor \frac{1}{2r} \rfloor$ ,  $\binom{Q}{h} \geq \delta n$ , as long as  $\delta \leq \frac{1}{4(h+1)(1+\frac{1}{2h})^2}$ . Let  $\varepsilon = h+1 - \frac{1}{2r}$ , and set  $\delta = 4\varepsilon$ .

We chose  $Q$  such that  $\binom{Q-1}{h} < \delta n$  and  $\binom{Q}{h} \geq \delta n$ . Then  $\binom{Q}{h} < \delta n \frac{Q}{Q-h} \leq \delta \frac{h+\frac{1}{2}}{h}$ , due to  $Q \geq 2h^2 - h$ . Hence  $Q \leq h \left( \delta n \frac{h}{h+(1/2)} \right)^{1/h}$ .

We upper bound the max latency of VBS in the following way.

$$\begin{aligned} L_{max} &\leq \max\{(h+1)(n-1) + Q(n-1), (h+1)(n-1) + (2h^2-h)(n-1)\} \\ &\leq 2(h+1)(n-1) + 2h^2(n-1) + h \left( 4\varepsilon n \frac{h+\frac{1}{2}}{h} \right)^{1/h} (n-1) \\ &\leq O(h[hN^{1/(h+1)} + (\varepsilon N)^{1/h}]) \end{aligned}$$

For sufficiently large  $N$  (determined by  $\varepsilon$  and  $h$ , both functions of  $r$ ), the second term will dominate. Thus, for large  $N$ :

$$L_{max} \leq O \left( h \left[ (\varepsilon N)^{1/h} + N^{1/(h+1)} \right] \right) = O(L^*(r, N)).$$

Note that by Lemma 4, VBS only gives a tight latency bound when  $4\varepsilon = \delta \leq \frac{1}{4(h+1)(1+\frac{1}{2h})^2}$ . However when  $\varepsilon$  is greater than this value, we can use EBS instead. By Lemma 3, EBS gives a factor  $C$  tight bound when  $\varepsilon > 2\sqrt{\frac{2h}{\pi}} \left( \frac{2e}{C} \right)^h$ . We check to make sure that there exists a constant  $C$  which works for all  $\varepsilon > \frac{1}{4} \cdot \frac{1}{4(h+1)(1+\frac{1}{2h})^2}$

$$\begin{aligned} 2\sqrt{\frac{2h}{\pi}} \left( \frac{2e}{C} \right)^h &\leq \frac{1}{4} \cdot \frac{1}{4(h+1) \left( 1 + \frac{1}{2h} \right)^2} \\ C &\geq O \left( \sqrt{h}^{1/h} \left( (h+1) \left( \frac{2h+1}{2h} \right)^2 \right)^{1/h} \right) = O(1) \end{aligned}$$

Since there exists such a factor  $C$ , the following holds for EBS in the regions of interest.

$$L_{max} \leq O \left( h \left[ (\varepsilon N)^{1/h} + N^{1/(h+1)} \right] \right) = O(L^*(r, N)) \quad \square$$

## 5 CONCLUSION AND OPEN QUESTIONS

In this paper we introduced a mathematical model of oblivious reconfigurable network design and investigated the optimal latency attainable for designs satisfying any given throughput guarantee,  $r$ . We proved that the best maximum latency achievable is  $\Omega(L^*(r, N))$ , for  $L^*(r, N) = h \left( N^{1/(h+1)} + (\varepsilon N)^{1/h} \right)$ . We also present two ORN designs, EBS and VBS. For every constant  $r$ , we show there exist infinitely many  $N$  for which either EBS or VBS achieves a maximum latency of  $O(L^*(r, N))$ .

Our investigation of the throughput-latency tradeoff for ORN designs affords numerous opportunities for follow-up work. In this section we sketch some of the most appealing future directions.

**Universal connection schedules** - EBS and VBS both use connection schedules tuned based on the targeted throughput rate  $r$ . Is there a single connection schedule that permits achieving the Pareto-optimal latency for a large range of  $r$ , or perhaps even for every value of  $r$ , merely by varying the routing scheme?

We conjecture that the following connection schedule, inspired by [24], supports ORN designs that are Pareto-optimal with respect to the tradeoff between worst-case throughput and *average* latency, for every value of  $r$ , when  $N$  is a prime power. Let  $\mathbb{F}$  denote the finite field with  $N$  elements, and let  $x$  denote a primitive root in  $\mathbb{F}$ . Define the sequence of permutations  $\pi_0, \pi_1, \dots$  by specifying that  $\pi_k(i) = i + x^k$  for all  $i \in \mathbb{F}$ ,  $k \in \mathbb{N}$ . We have experimented with this family of connection schedules when  $\mathbb{F}$  is a prime field and 2 is a

primitive root, for values of  $N$  ranging from 11 up to around 300. We numerically verified that in all cases we tested, for each value of  $r$  ranging from  $\frac{1}{2}$  down to roughly  $\frac{1}{\log n}$ , there is an oblivious routing scheme guaranteeing throughput  $r$ , whose average latency is within a constant factor of matching our lower bound. In fact, the average latency in most cases that we tested was moderately less than EBS's. However, thus far we have not succeeded in proving that this pattern persists for infinitely many  $N$ .

**Bridging the gap between theory and practice** - Our model of ORNs uses idealized assumptions that gloss over important details that affect the performance of ORNs in practice. A more realistic model would not equate expected congestion with actual congestion. This would necessitate grappling with the issues of queueing and congestion control. It also opens the Pandora's box of non-oblivious routing, since a frame that was intended to be transmitted on link  $(u, v)$  but finds that link blocked due to congestion must either be transmitted in a different timeslot, or on a different link in the same timeslot; in either case the frame's path in the virtual topology differs from the intended one. An appealing middle ground between fully centralized control (as in classical models of circuit-switched networks) and a fully oblivious model (as in our paper) could be a network design with a fully oblivious connection schedule coupled with a partially-adaptive routing scheme based on local information such as queue lengths at traversed nodes.

Our model also fails to account for (possibly heterogeneous) propagation delays. The model could be enhanced to take propagation delay into account by adjusting the virtual topology. Rather than connecting physical edges from  $(i, s)$  to  $(j, s + 1)$ , they could instead connect to  $(j, s + d_{ij})$ , where  $d_{ij}$  is a whole number representing the propagation delay from  $i$  to  $j$  in units of timeslots. As in our basic model, nodes of the virtual topology in this enhanced model would be constrained to belong to at most one incoming and at most one outgoing physical edge, though if  $d_{ij}$  varies with  $i$  and  $j$  then the set of physical edges would no longer be described by a sequence of permutations.

**Supporting multiple traffic classes** - In this paper we sought to optimize the worst-case latency for network designs that guarantee a specified rate of throughput. In practice, flows co-existing on a network can differ markedly in their latency sensitivity. Can EBS, VBS, or other ORN designs be adapted to simultaneously offer users a menu of options targeting different points on the latency-throughput tradeoff curve? What guarantees can such network designs provide to the different classes of traffic they serve?

## ACKNOWLEDGEMENTS

This work was supported in part by NSF grants CCF-1512964, CSR-1704742, CNS-2047283, and DBI-2019674, a Google faculty research scholar award, a Microsoft Investigator Fellowship, and a Sloan fellowship. Part of this work was done while some authors were visiting the University of Washington.

## REFERENCES

- [1] Daniel Amir, Tegan Wilson, Vishal Shrivastav, Hakim Weatherspoon, Robert Kleinberg, and Rachit Agarwal. 2021. Optimal Oblivious Reconfigurable Networks. *CoRR* abs/2111.08780 (2021). <https://doi.org/10.48550/arXiv.2111.08780>
- [2] David L. Applegate and Edith Cohen. 2003. Making intra-domain routing robust to changing and uncertain traffic demands: understanding fundamental tradeoffs. SIGCOMM '03. <https://doi.org/10.1145/863955.863991>
- [3] Yossi Azar, Edith Cohen, Amos Fiat, Haim Kaplan, and Harald Räcke. 2003. Optimal Oblivious Routing in Polynomial Time. STOC '03. <https://doi.org/10.1145/780542.780599>
- [4] Hitesh Ballani, Paolo Costa, Raphael Behrendt, Daniel Cletheroe, Istvan Haller, Krzysztof Jozwik, Fotini Karinou, Sophie Lange, Kai Shi, Benn Thomsen, et al. 2020. Sirius: A Flat Datacenter Network with Nanosecond Optical Switching. SIGCOMM '20. <https://doi.org/10.1145/3387514.3406221>
- [5] Marcin Bienkowski, Mirosław Korzeniowski, and Harald Räcke. 2003. A Practical Algorithm for Constructing Oblivious Routing Schemes. SPAA '03. <https://doi.org/10.1145/777412.777418>
- [6] Cheng-Shang Chang, Duan-Shin Lee, and Yi-Shean Jou. 2002. Load balanced Birkhoff-von Neumann switches, part I: one-stage buffering. *Computer Communications* 25, 6 (2002), 611–622. [https://doi.org/10.1016/S0140-3664\(01\)00427-3](https://doi.org/10.1016/S0140-3664(01)00427-3)
- [7] Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. 2004. A tight bound on approximating arbitrary metrics by tree metrics. *J. Comput. Syst. Sci.* 69, 3 (2004). <https://doi.org/10.1145/780542.780608>
- [8] Nathan Farrington, George Porter, Sivasankar Radhakrishnan, Hamid Hajabdolali Bazzaz, Vikram Subramanya, Yeshaiah Fainman, George Papen, and Amin Vahdat. 2010. Helios: a hybrid electrical/optical switch architecture for modular data centers. SIGCOMM '10. <https://doi.org/10.1145/1851275.1851223>
- [9] 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. SIGCOMM '16. <https://doi.org/10.1145/2934872.2934911>
- [10] Anupam Gupta, Mohammad Taghi Hajiaghayi, and Harald Räcke. 2006. Oblivious network design. SODA '06.
- [11] Mohammad Taghi Hajiaghayi, Robert D. Kleinberg, Frank Thomson Leighton, and Harald Räcke. 2006. New lower bounds for oblivious routing in undirected graphs. SODA '06.
- [12] Chris Harrelson, Kirsten Hildrum, and Satish Rao. 2003. A polynomial-time tree decomposition to minimize congestion. SPAA '03. <https://doi.org/10.1145/777412.777419>
- [13] Prahladh Harsha, Thomas P. Hayes, Hariharan Narayanan, Harald Räcke, and Jaikumar Radhakrishnan. 2008. Minimizing average latency in oblivious routing. SODA '08.
- [14] Christos Kaklamani, Danny Krizanc, and Thanasis Tsantilas. 1991. Tight Bounds for Oblivious Routing in the Hypercube. *Math. Syst. Theory* 24, 4 (1991), 223–232. <https://doi.org/10.1145/97444.97453>
- [15] Jongman Kim, Chrysostomos Nicopoulos, Dongkook Park, Reetuparna Das, Yuan Xie, Vijaykrishnan Narayanan, Mazin S. Yousif, and Chita R. Das. 2007. A Novel Dimensionally-Decomposed Router for on-Chip Communication in 3D Architectures. ISCA '07. <https://doi.org/10.1145/1273440.1250680>
- [16] Praveen Kumar, Yang Yuan, Chris Yu, Nate Foster, Robert Kleinberg, Petr Lapukhov, ChiuLin Lim, and Robert Soulé. 2018. Semi-Oblivious Traffic Engineering: The Road Not Taken. NSDI '18.
- [17] He Liu, Feng Lu, Alex Forencich, Rishi Kapoor, Malveeka Tewari, Geoffrey M. Voelker, George Papen, Alex C. Snoeren, and George Porter. 2014. Circuit Switching Under the Radar with REACToR. NSDI '14.
- [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. NSDI '20.
- [19] 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. SIGCOMM '17. <https://doi.org/10.1145/3098822.3098838>
- [20] George Porter, Richard Strong, Nathan Farrington, Alex Forencich, Pang Chen-Sun, Tajana Rosing, Yeshaiah Fainman, George Papen, and Amin Vahdat. 2013. Integrating Microsecond Circuit Switching into the Data Center. SIGCOMM '13. <https://doi.org/10.1145/2486001.2486007>
- [21] H. Räcke. 1975. Minimizing congestion in general networks. FOCS '75. <https://doi.org/10.1109/SFCS.2002.1181881>
- [22] Harald Räcke. 2008. Optimal Hierarchical Decompositions for Congestion Minimization in Networks. STOC '08. <https://doi.org/10.1145/1374376.1374415>
- [23] 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. NSDI '19.
- [24] Edward Tremel, Ken Birman, Robert Kleinberg, and Márk Jelasity. 2019. Anonymous, Fault-Tolerant Distributed Queries for Smart Devices. *ACM Trans. Cyber Phys. Syst.* 3, 2 (2019), 16:1–16:29. <https://doi.org/10.1145/3204411>
- [25] Leslie G. Valiant. 1982. A Scheme for Fast Parallel Communication. *SIAM J. Comput.* 11, 2 (1982), 350–361. <https://doi.org/10.1137/0211027>
- [26] Leslie G Valiant and Gordon J Brebner. 1981. Universal schemes for parallel communication. STOC '81. <https://doi.org/10.1145/800076.802479>
- [27] 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. SIGCOMM '10. <https://doi.org/10.1145/1851275.1851222>