Reducing Waste in Data Center Network Upgrades

Andrew R. Curtis, S. Keshav, and Alejandro Lopez-Ortiz
Cheriton School of Computer Science
University of Waterloo

University of Waterloo Technical Report CS-2010-08

ABSTRACT

Fundamental limitations of traditional data center network architectures have led to the development of architectures that provide enormous bisection bandwidth for up to hundreds of thousands of servers. To implement any of these solutions, a legacy data center operator usually must replace most switches in their network—a tremendously wasteful proposition in terms of capital costs, energy used in manufacturing, and raw materials. To enable such upgrades, we introduce a construction for the first data center network topology that supports heterogeneous switches while providing guarantees on bisection bandwidth and using optimal link capacity. We finish by describing an algorithm that finds the minimum cost upgrade path for an existing network to support a given traffic load.

1. INTRODUCTION

Most current data center networks use 1+1 redundancy in a three-level tree topology, which provides inadequate bisection bandwidth, reducing data center agility. This reduces server utilization when workloads vary rapidly because dynamic reallocation of services to servers is impossible. Recent work has addressed this problem by providing enormous bisection bandwidth for up to hundreds of thousands of servers [1, 7, 9, 10, 18]. However, these solutions assume homogeneous switches, each with a prescribed number of ports. Therefore, adopting these solutions comes at the cost of replacing nearly all switches in their network. This is not only wasteful but also usually infeasible due to sunk capital costs, downtime, and time to market.

The goal of our work is to allow a data center operator to incrementally add equipment to boost bisection bandwidth and reliability without needing to throw out the entire existing network. Adding switches rather than replacing them greatly reduces cost as well as the environmental impact of a data center. However, this results in the creation of a heterogeneous tree-like topology, a topology that has not been sufficiently studied in past work. Therefore, we provide the theoretical foundations for the analysis of heterogeneous Clos networks. We also use these results to construct an algorithm we call OptUp that allows data center operators to determine the minimal cost upgrade path to meet given traffic constraints. Our construction is provably optimal in that it uses the minimal amount of link capacity possible to meet these traffic constraints. To our knowledge this is the first construction that achieves this optimality while supporting switches with heterogeneous rates and numbers of ports.

Our key contributions are:

- Analysis of the relative dollar and energy costs of reusing versus replacing networking infrastructure in a data center.
- A theoretical basis for constructing optimal heterogeneous Clos topologies that support routing all hose traffic matrices.
- The OptUp algorithm to determine the minimum set of networking equipment that needs to be purchased to upgrade an existing topology to meet a given oversubscription guarantee.

The rest of this paper is as follows. We begin with a background on data center topologies in Section 2. In Section 3 we compare the costs of replacing and reusing networking infrastructure in a data center. Section 4 provides the theoretical foundations for the analysis of heterogeneous Clos networks. The OptUp algorithm is described in Section 5. We end with a discussion of our work and conclusions in Section 6.

2. BACKGROUND AND RELATED WORK

Most current DCN switching fabrics form a multi-rooted tree. Servers are arranged in racks—each rack typically contains 40–80 servers—and servers connect to top-of-rack (ToR) switches, which usually have 48 1 Gb and 2–4 10 Gb ports. These ToR switches are the leaves of the multi-rooted switching tree. This tree usually has three levels: the ToR switches connect to a level of aggregation switches which connect to a core level made up of either switches or routers. The core level is connected to the Internet using edge routers.

Model	No. Ports	Rates (Gb)	Watts	Price	Energy cost/yr	Years before energy cost dominants
Arista 7048	48, 4	1, 10	235	$11,995	$206	58
Arista 7124S	24	10	288	$13,195	$252	52
Juniper EX2500	24	10	190	$18,000	$166	108
Arista 7148S	48	10	588	$20,895	$515	40
Juniper EX8216	128	10	10k	$716k	$8,760	81

Table 1: Capital cost vs. energy cost for a few switches. We assume a relatively high price of 10 cents per kWh for electricity and that the switch uses its maximal rated power 24 hours/day. Power draw information is from the vendor's product information.

This architecture has two major problems—poor reliability and insufficient bisection bandwidth—besides many other minor problems, as detailed by Greenberg *et al.* [7, 8].

These limitations have been the focus of much recent work and researchers have proposed a variety of topology constructions. Some current DCN proposals are based on classic network topologies such as fat-trees [1], the Clos network (a generalization of fat-trees) [7], and hypercubes [18]. Others employ recursive constructions [9, 10]. These proposals, however, have a common feature: they are highly regular and require homogeneous switches, each with a prescribed number of ports. This makes it nearly impossible to implement them as an upgrade to an existing data center without replacing most switches in the network. Further, the proposed topologies all take a one size fits all approach: they do not allow operators flexibility in determining bandwidth requirements of different rows in the data center, instead all rows must have uniform uplink bandwidth.

The most closely related topology construction to our work here is that of Rasala and Wilfong [15], who gave a strict nonblocking construction for networks with heterogeneous switches. Our work differs in two key aspects. First, they dealt with strictly nonblocking networks whereas our constructions are rearrangeably nonblocking in their setting, so our construction require much less link capacity. Second, their constructions only connect switch sets with two types of switches and they do not support heterogeneous switch port speeds. Our construction supports any number of switch types and port speeds.

3. COMPARING THE COSTS OF REPLACING VERSUS REUSING NETWORKING EQUIPMENT IN A DATA CENTER

We now compare the capital, operational, and environmental cost of upgrading a DCN and contrast it with the cost of replacing it with all-new infrastructure. The capital cost of the network is a significant portion of a data center's overall operating budget. Recent estimates peg the capital cost of networking equipment—switches, routers and load balancers—at around 13–20% of the total monthly budget of a 50,000 server data center [8, 11], assuming data center operates with a PUE[1] of 1.7 and a three year amortization period for servers and network equipment. Lengthening the life of networking equipment can therefore tangibly reduce a data center's operational cost; for example, the yearly cost of our example data center is over $17 million, so doubling the life of all existing networking equipment can save up to $7.65 million over three years and reduces the data center's operating budget by 7.5%. This indicates our approach is at least promising. We now explore this saving at greater depth.

One reason to replace equipment is to save energy costs, e.g., by replacing several inefficient devices with a new device that draws much less power. To examine this approach in the context of a DCN, Table 1 lists details of a few switches. As the table shows, the expense of a switch is dominated by capital cost, not energy cost, making it unlikely that replacing old switches with a new switch will save money overall. For example, replacing 10 Arista 7048 switches with a 48 port 10 Gb switch saves $2,192 per year on energy costs assuming PUE of 1.7 and 10 cents per kWh, but it still takes over 9.5 years for the power savings to pay for the capital costs of the new switch. Therefore, replacing switches on the basis of energy savings does not make sense currently.

The operational power draw of a switch is only one component of its total energy use—a complete analysis should include the energy used to manufacture it. We are not aware of any such analysis for networking equipment; however, studies have been conducted to determine the amount of energy used to manufacture a personal computer, which should be on the same order of magnitude as the energy used producing a 1U switch. Williams has estimated that over 2 MWh of energy is used to produce a desktop PC [17]. If we take this number as an approximation for producing a switch, we find that manufacturing a switch consumes as much energy as powering that switch for 6–12 months for 24 hours/day. When a switch like the Arista 7048 has a three year lifetime, manufacturing it accounts for nearly

[1]Power Usage Efficiency (PUE) is the total facility power/IT equipment power. An average DC has a PUE of 2.0 [4], and the industry best is 1.2 [6].

25% of its total energy use. Therefore, doubling the lifetime of switches can reduce their lifetime energy consumption by up to 12.5%.

The downside of extending switches lifetimes is that, as hardware ages, its likelihood of failure increases. Every time a switch (or component of a switch such as a port or line card) fails, it impacts the network's ability to meet demand and necessitate human intervention. Given a traditional DCN with only 1+1 redundancy, the a switch failure seriously degrades network performance. However, if the network has been upgraded according to our approach, the impact of a single switch failure is minimal—the network will have N+N redundancy, so removing a single switch has minimal impact on the amount of bisection bandwidth available. A failed switch will need to be replaced by a human; however, these replacements will have to be performed regardless, and most switches will operate long past their amortization period, so IT staff workload will not be adversely effected by our approach.

4. OUR APPROACH

We are interested in upgrading legacy DCNs, so our approach must be applicable to the traditional 1+1 topology, which happens to be a form of a Clos network. However, Clos networks cannot make use of heterogeneous switches as we described previously. Therefore, we focus on the analysis of heterogenous Clos networks. To give optimal results, our construction places strict constraints on the number of ports available at switches in the core level of the network, so it may not be not applicable in every situation.

Few detailed studies of data center traffic have been published; however, the studies to date demonstrate that DCNs exhibit highly variable traffic [2, 7, 12]—the traffic matrix (TM) in a DCN shifts frequently and its overall volume changes dramatically in short time periods. To account for this, our constructions can feasibly route, i.e., no link's utilization is ever higher than 1, all TMs that are possible given a maximum ingress/egress rate for each node is known as the *hose traffic matrices* [3, 5]. We denote the rate of each node by $r(i)$.

To simplify our analysis, we use multipath routing, i.e., a TM D represents a multi-commodity flow and entry D_{ij} is a demand from i to j. Modeling routing as a network flow is not entirely realistic because a single flow can be split fractionally across multiple paths. However, we expect our theoretical results to be indicative of the behavior of a practical implementation because long-lived flows are rare in DCN traffic [7]. We address this issue further in Section 4.3.

Before presenting our heterogeneous Clos construction in Section 4.2, we briefly review the Clos network.

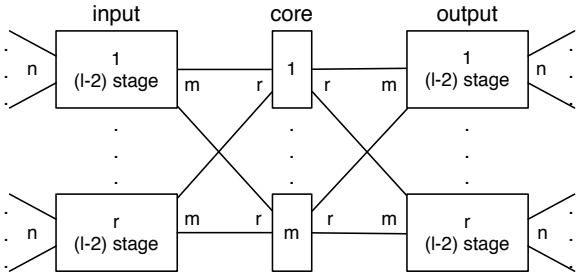

Figure 1: Unfolded l-stage Clos network. Each input and output switch here is a subnetwork with $l-2$ stages.

4.1 The Clos network

A 3-stage Clos network, denoted by $C(n, m, r)$, is an interconnection network where the first stage, made up of input switches, consists of r switches, each with n inlets and m uplinks. Symmetrically, the third stage consists of r output switches, each with n outlets and m downlinks. The second stage then is m switches, each with r links to first-stage switches and r links to third-stage switches. We call the switches in the middle stage the *core switches*. We refer to the links from a stage to a higher stage as *uplinks* and the links from a stage to a lower stage as *downlinks*. A *folded Clos network* places input and output layers top of each other; only bidirectional networks can be folded. DCNs are folded Clos networks, so we use the folded Clos variant in this paper, and when doing so, the input and output switches are the same devices, so we refer to them as input/output (IO) switches.

The recursive nature of Clos network means that we only have to deal with 3-stage Clos networks. An l-stage Clos network is recursively composed of 3-stage Clos networks. In an l-stage Clos network, each input and output switch is replaced by an $(l-2)$-stage network. An example is shown in Figure 1. As a result of this recursive construction, any algorithm or theorem that applies to a 3-stage Clos network applies to an l-stage Clos networks by applying it to the outermost 3-stage network first, and then recursively applying it to the $(l-2)$-stage subnetworks. As such, we always deal with 3-stage networks in this paper, but our results can be generalized to an l-stage Clos networks in a straightforward manner.

4.2 Constructing A Heterogeneous Clos Network

We separate the logical network design (§4.2.1) from the problem of finding a physical realization (§4.2.2).

4.2.1 Logical design

The logical topology of a Clos network $C(n, m, r)$ col-

lapses all core switches into a single logical node, so the logical topology of $C(n, m, r)$ is a tree. This logical tree's root node has r children—the r input/output switches—and the tree's leaves represent inlets and outlets. Here, an edge between an IO node and the root represents m links in the underlying physical realization. For the remainder of this section, we are concerned with the design of logical topologies that use the minimal link capacity necessary and sufficient to feasibly route all hose TMs (i.e., the logical topology is optimal), and we make the assumption that a logical node can be realized using the same amount of switching capacity as the logical topology. We lift this assumption in the next section when we show how to find such physical realizations.

Note that in our construction, switches need not uniformly have n inlets and outlets. We let each IO switch i have a rate, denoted by $r(i)$, which is the sum of its downlink rates (e.g., in a homogeneous network, the rate of each IO switch is n). Each logical edge (i, x) between an IO node i and logical core node x has a capacity $c(i, x)$, which is the sum of physical link rates that (i, x) represents. A logical topology has *optimal edge capacity* if the sum of edge capacities is equal to the sum of node rates.

We are now ready to give our main logical design results. The following characterizes logical arrangements that use minimal link capacity to feasibly route all hose TMs.

LEMMA 1. *Let T be a logical topology with input/output nodes $I = \{1, \ldots, k\}$, and let $x_1, \ldots, x_l$ be the root nodes of T. Let X_p denote the set of input/output nodes neighboring root node x_p such that $X_1 = I$ and $X_1 \supset \cdots \supset X_l$. Whenever all edges of T have positive capacity, we have that T feasibly routes all hose TMs with optimal edge capacity if, for all x_p, such that $2 \leq p \leq l$,*

$$r(i) > \sum_{j \in X(x_{p-1}) - X(x_p)} r(j) \quad \text{for all } i \in X_p \tag{1}$$

and $|X_l - X_{l-1}| \geq 2$.

PROOF. Suppose there is some logical topology T that has a root node x such that there is a node $i \in X_{l'}$, where l' is the maximal root node i neighbors, with $c(i, x_{l'}) > 0$ and for which Equation 1 does not hold. Consider how much capacity the edges $(i, x_1), \ldots, (i, x_{l'-1})$ must have since T can serve all hose TMs: there must be at least $\min\{r(i), \sum_{j \in X_1 - X_{l'}} r(j)\}$ capacity to these nodes otherwise there is a hose TM that T cannot feasibly route. By assumption, $r(i) \leq \sum_{j \in X_1 - X_{l'}} r(j)$, so $r(i)$ is the minimal here. In a logical topology with optimal edge capacity, each IO node has at most $r(i)$ of uplink capacity. However, here, we have that i has $r(i) + c(i, x_{l'}) > r(i)$ uplink capacity, contradicting the optimality of T.

Suppose that $|X_l - X_{l-1}| = 1$. Here, T is non-optimal since the root node x_l has only a single neighbor, so it cannot route traffic to any other IO nodes. Therefore, it should not have positive capacity, since all traffic will need to be routed through $x_1, \ldots, x_{l-1}$ anyhow. □

The following results are implied by this lemma:

- whenever $r(1) = \cdots = r(k)$, the optimal logical topology has a single root node, and
- no matter the rates of each IO node, a logical topology with a single root node is optimal, i.e., a logical topology can always use fewer root nodes than it's allowed by Lemma 1 and be optimal.

This lemma identifies the available logical topologies for a set I of IO nodes, but it does not determine the capacities of each logical edge. The following theorem shows how capacity can be assigned to the logical edges of T to feasibly route all hose TMs. The intuition underlying this theorem is that the root x_p and its children (the IO switches) form a disjoint spanning tree. We provision the spanning tree rooted at x_1 first, and then move to the next root node's spanning tree. Every unit of capacity that is provisioned to x_1 is a unit that does not have to be routed through $x_2, \ldots, x_l$, so we subtract off the previously allocated capacity from the edges to $x_2, \ldots, x_l$.

THEOREM 2. *Let T be a logical topology with input/output nodes $I = \{1, \ldots, k\}$, and let $x_1, \ldots, x_l$ be the root nodes of T such that T has an optimal number of root nodes by Lemma 1. Let X_p denote the set of input/output nodes neighboring root node x_p such that $X_1 = I$ and $X_1 \supset \cdots \supset X_l$, and let $X_0 = \emptyset$ and $X_{l+1} = \emptyset$. We have that T can feasibly route all hose TMs using optimal capacity if and only if*

$$c(i, x_p) = \begin{cases} \sum_{j \in X_p - X_{p+1}} r(j) & \text{if } i \in X_{p+1}, \\ r(i) - \sum_{j \in I - X_p} r(j) & \text{otherwise} \end{cases} \tag{2}$$

for all $1 \leq p \leq l$ and all $i \in I$.

PROOF. Suppose that T can feasibly route all hose TMs and that Equation 2 holds for all edges except (i, x_p). Let l' be the maximum root node such that $i \in X(x_{l'})$. Because T can feasibly route all hose TMs, we have:

$$\begin{aligned} \sum_{u \in [l']} c(i, x_u) &= \sum_{q \in [l'-1]} \sum_{j \in X_q - X_{q+1}} r(j) \\ &\quad + r(i) - \sum_{j \in X_1 - X_{l'}} r(j) && (3) \\ &= \sum_{j \in X_1 - X_{l'}} r(j) + r(i) - \sum_{j \in X_1 - X_{l'}} r(j) && (4) \\ &= r(i) && (5) \end{aligned}$$

So, we have

$$\sum_{q\in[l'-1]}\sum_{j\in X_q - X_{q+1}} r(j) = \sum_{j\in X_1 - X_{l'}} r(j) \qquad (6)$$

whenever T can feasibly route all hose TM with minimal edge capacity. However, here, we find a contradiction in both possible cases.
Whenever $i \in X(x_{p+1})$, so $c(i,x_p) < \sum_{j\in X_{p-1}-X_p} r(j)$, the left hand side of Equation 6 is less than the right hand side. And otherwise, $c(i,x_p) < r(i) - \sum_{j\in X_1 - X_p} r(j)$, in which case, we cannot make the reduction from Equation 4 to Equation 5.

To show sufficiency, suppose that Equation 2 holds for all edges of T. We construct a multipath routing that feasibly routes any hose TM D_{ij}. Let $i, j \in I$ be IO nodes such that $r(i) \le r(j)$ and let l' be the max root node where $i, j \in X_{l'}$. When sending to j, let i split its traffic across root nodes $x_1, \ldots, x_{l'}$ such that $D_{ij}/c(i,x_p)$ traffic is routed through x_p, for $1 \le p \le l'$, and then x_p forwards this traffic to j on its single edge to j. For any hose TM D, the max traffic i can send is $r(i)$, so the max traffic i places on edge (i,x_p) is $r(i)/c(i,x_p)$. Since Equation 2 holds for all edges, we have that

$$\sum_{u\in[l']} c(i,x_u) = r(i)$$

as established in Equations 3–5 above. Therefore, i can send traffic at rate up to $r(i)$ and never overload a link. Similarly, i cannot overload a link while receiving traffic, because it cannot receive more than $r(i)$ traffic at once. □

We give two examples of an optimally provisioned logical topology in Figure 2. This theorem prescribes the amount of capacity needed in a logical topology, yet it is flexible in assignment of this capacity across logical edges. This is beneficial because the physical constraints of switches make many logical topologies infeasible to construct in practice.

4.2.2 Physically realizing a logical node

We now show how to find a physical realization of a logical node. Here, we are given a logical core node and a set of IO switches, and we want to find a set of switches that realizes the core node.

Each IO switch has a set of uplink ports, which may have multiple speeds. To simplify our presentation, we separate IO nodes with multiple uplink port speeds into separate switches, so that each IO switch has a single uplink port speed. This does not lead to a loss of generality because we can recombine the separated switches later. So, each IO switch i has an single uplink port speed, denoted by $p(i)$. We assume that an IO switch i has at least $\lceil r(i)/p(i) \rceil$ ports; otherwise, no realization that can feasibly route all hose TMs exists.

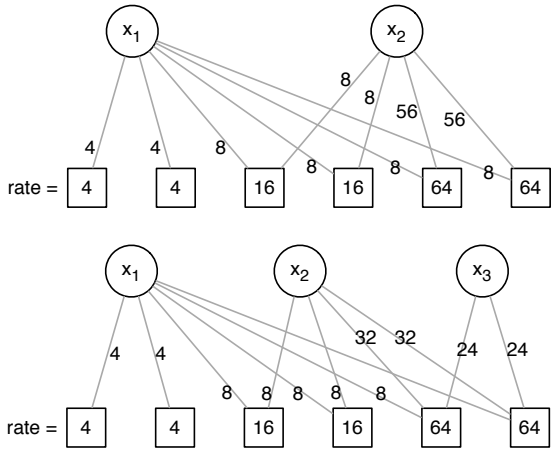

Figure 2: Two optimal logical configurations for a set of IO switches. The uplink rate of each IO switch is shown inside of the switch, and the numbers by the edges indicate the edge's capacity. Despite the different number of core nodes, each of these logical topologies is optimal.

We now show how to realize a logical core node x with a set I of IO switches as its children. We use X to denote the set of switches that makeup logical node x. Let $m(i) = \lceil c(i,x)/p(i) \rceil$, where $c(i,x)$ is the capacity of the logical edge (i,x) as before. Here, $m(i)$ is the number of physical uplinks i has to x. We use $P(r)$ to denote the set of all switches of I with $p(i) = r$, and $I(x)$ denotes the set of IO switches neighboring root x.

Now, we need to determine how many switches are in X and how many ports each has. Let $m_{\min} = \min_{j\in I(x)}\{m(j)\}$. The core switches that realize x and the IO switches $I(x)$ form a complete bipartite graph, so we have $|X| = m_{\min}$. Each core switch in X must have at least $m_{\min} \cdot |P(r)|$ ports with speed r, for each port speed r, and each $i \in I(x)$ has $\lceil m(i)/m_{\min} \rceil$ uplinks to each switch in X.

THEOREM 3. *A physical realization G constructed as described above of a logical tree T with root node x and input/output switches I with $c(i,x)$ minimized according to Theorem 2 can feasibly route all hose TMs.*

Further, if $c(i,x)$ and $m(i)$ are evenly divisible by $p(i)$ and $m_{\min}$ respectively for all $i \in I$, then the amount of link capacity used by this physical realization matches the lower bound of any switching network that can feasibly route all hose TMs.

PROOF. To show that G can feasibly route all hose TMs, by Theorem 2, it's enough to show that there is a routing which distributes $r(i)/c(i,x)$ traffic over the physical links of the logical edges (i,x) and (x,i) without overloading any physical links. When i sends traffic to x, let each physical uplink carry $p(i)/c(i,x)$

fraction of the traffic, no matter the destination, and then the receiving core switch forwards the traffic to its destination. Then any traffic matrix can be handle as long as i never sends more than:

$$r(i) \cdot \sum_{v \in X} p(i)/c(i,x)\lceil m(i)/m_{\min} \rceil =$$
$$r(i) \cdot |X| \Big(\Big\lceil \frac{c(i,x)}{m(i)} \Big\rceil / c(i,x) \cdot \lceil m(i)/m_{\min} \rceil \Big) =$$
$$r(i) \cdot m_{\min} \big(1/m_{\min}\big) =$$
$$r(i)$$

traffic, which i will never exceed in a hose TM. By a similar argument, there is enough link capacity from the physical switches in X to i.

An optimal construction has a total link capacity of $2\sum_{i \in I} r(i)$. To see that the construction above matches this bound when $c(i,x)$ and $m(i)$ are evenly divisible by $p(i)$ and $m_{\min}$ respectively for all $i \in I$, consider the above equations. In this case, each $i \in I$ has $r(i)$ uplink capacity and $r(i)$ downlink capacity. Summing this over all switches in I shows our construction is optimal. □

In the above theorem we claim that our construction needs only as much link capacity as any other switching network that can feasibly route all hose TMs. A switching network is a network where nodes with positive rate (i.e., $r(i) > 0$) never directly connect to other nodes with positive rate, that is, all nodes connect to switches. A corollary to a result of Zhang-Shen and McKeown [19] is that any switching network with node rates $r(1), \ldots, r(n)$ can feasibly route all hose TMs iff the total link capacity is at least $\sum_{1 \le i \le n} 2r(i)$. This bound is matched by, for example, a homogeneous 3-stage Clos network when all IO switch rates are equal. Our construction matches this bound without any restrictions on IO switch rates.

4.3 Routing

The theorems in the previous section state that a feasible routing of any hose TM exists on our heterogeneous Clos network construction; however, we have not described how to find such a routing. To prove these theorems, we used multipath routing, where a flow can be split across multiple paths. This approach is not practical, however, because switches do not currently support splitting a flow across multiple paths. Even if switches did support flow splitting, out of order packet arrival could cause problems with TCP.

For practical routing our heterogeneous constructions, we offer two possibilities. First, one could use Mudigonda *et al.*'s SPAIN [14], a recent system to load balance DCN traffic over arbitrary topologies. SPAIN works by created many VLANs over a network topology, each one a spanning tree, and then spreading flows over these spanning trees. As our constructions give tree-like topologies, SPAIN would be to extract the full bisection bandwidth from our constructions provided that flows can be effectively load balanced across VLANs.

The second routing approach is to load balance flows using oblivious routing, that is, the path for an i-j flow is randomly selected from a probability distribution over all i-j paths available. Valiant load balancing (VLB) [16] is one example of oblivious routing, and VLB provides optimal routing on a Clos network when flows can be split across multiple paths (packet-level VLB). Flow-level VLB has been tested as a load balancing mechanism by Greenberg *et al.* on a Clos network, and was found to balance traffic with near optimal results [7], namely because long lived flows are rare in the data center. In their experiments, flow-level VLB performed within 94% of optimal. Packet-level VLB is no longer optimal on our heterogeneous Clos networks; however, there is a packet-level optimal oblivious routing on our constructions, and the routing can be found using linear programming [13].

THEOREM 4. *When flows can be split across multiple paths, there exists an oblivious routing that can feasibly route all hose TMs on a heterogeneous Clos network that meets the conditions of Theorem 3.*

PROOF. In the proof of Theorems 2 and 3, we showed that a feasible multipath routing exists for any hose TM on a heterogeneous Clos network. As the same routing is feasibly for any hose TM, it is an oblivious routing for all hose TMs. □

Even though we cannot perform packet-level oblivious routing in practice, previous positive results regarding flow-level VLB lead us to believe that flow-level oblivious routing will perform well on a heterogeneous Clos network as well.

5. THE OPTUP ALGORITHM

We now sketch the OPTUP algorithm. As input, it takes a description of the data center's infrastructure and the prices of available switches. OPTUP returns a min-cost set of switches to purchase and an upgraded network topology.

OPTUP does the following:

1. generate all feasible logical topologies as determined by Lemma 1, and then
2. find a min-cost physical realization of each logical topology.

Step 1 is easily done by enumeration because the number of feasible logical topologies grows linearly with the number of IO nodes.

Step 2 is an NP-hard problem, which we solve using a branch and bound algorithm. Branch and bound is a general algorithm to solve optimization problems, and it

enumerates the solution space efficiently by eliminating branches of solutions known to be non-optimal.

By Theorem 3, we can take our solution space to be the set of IO switches because the core switches are determined by the IO switches (i.e., a heterogeneous Clos network is uniquely defined by the IO switch set). So, each node in the search tree is labeled by a switch type, and we branch a node by giving it children labeled with the available switch types. Each search tree node is a partial solution, representing the set of IO switches denoted by the labels on nodes to its path to the search tree's root.

Branch and bound trims branches of the search tree by lower bounding the cost of partial solutions, and discarding the subtree of a partial solution that cannot contain an optimal solution. Here, our lower bound for a partial solution is equal to the cost of partial solution's label plus an estimate of additional IO switch cost plus an estimate of core switches cost. The additional IO switch cost estimate is found by greedily adding a min-cost switch from the available types until the solution has enough capacity to satisfy the requirements of Theorem 2. The core switch cost is estimated by summing the number of ports/speed multiplied by the number of ports of that rate needed as dictated by Theorem 3. We can also use Theorem 3 to determine when a partial solution solution is not feasible and then trim its branch.

We are currently evaluating the performance of our algorithm and plan to report its performance in future work.

6. CONCLUSION

In this paper, we have argued that upgrading a DCN by adding switches, rather than by replacing switches significantly lowers capital costs and reduces a data center's overall energy burden. To enable such upgrades, we gave theoretical results for constructing DCNs with heterogeneous switches. Finally, we described the OptUp algorithm which can find a minimum cost upgrade path so an existing network can feasibly route a set of hose TMs. Together, our results here are a step towards a DCN architecture where the existing network does not have to be replaced to increase bisection bandwidth.

7. REFERENCES

[1] M. Al-Fares, A. Loukissas, and A. Vahdat. A scalable, commodity data center network architecture. In *ACM SIGCOMM*, 2008.

[2] T. Benson, A. Anand, A. Akella, and M. Zhang. Understanding data center traffic characteristics. In *Proceedings of the 1st ACM workshop on Research on enterprise networking (WREN)*, 2009.

[3] N. G. Duffield, P. Goyal, A. Greenberg, P. Mishra, K. K. Ramakrishnan, and J. E. van der Merive. A flexible model for resource management in virtual private networks. In *ACM SIGCOMM*, 1999.

[4] EPA. Epa report to congress on server and data center energy efficiency. Technical report, U.S. Environmental Protection Agency, 2007.

[5] J. A. Fingerhut, S. Suri, and J. S. Turner. Designing least-cost nonblocking broadband networks. *J. Algorithms*, 24(2):287–309, 1997.

[6] Google. Efficient computing–step 2: efficient datacenters. `http://www.google.com/corporate/green/datacenters/step2.html`.

[7] A. Greenberg, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. Maltz, P. Patel, and S. Sengupta. VL2: a scalable and flexible data center network. In *ACM SIGCOMM*, 2009.

[8] A. G. Greenberg, J. R. Hamilton, D. A. Maltz, and P. Patel. The cost of a cloud: research problems in data center networks. *Computer Communication Review*, 39(1):68–73, 2009.

[9] C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang, and S. Lu. BCube: a high performance, server-centric network architecture for modular data centers. In *ACM SIGCOMM*, 2009.

[10] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, and S. Lu. Dcell: a scalable and fault-tolerant network structure for data centers. In *ACM SIGCOMM*, 2008.

[11] J. R. Hamilton. Data center networks are in my way. Presented at the Standford Clean Slate CTO Summit, 2009.

[12] S. Kandula, S. Sengupta, A. Greenberg, and P. Patel. The nature of datacenter traffic: Measurements & analysis. In *ACM IMC*, 2009.

[13] M. Kodialam, T. V. Lakshman, and S. Sengupta. Maximum throughput routing of traffic in the hose model. In *IEEE Infocom*, 2006.

[14] J. Mudigonda, P. Yalagandula, M. Al-Fares, and J. C. Mogul. SPAIN: COTS data-center ethernet for multipathing over arbitrary topologies. In *USENIX NSDI*, 2010.

[15] A. Rasala and G. Wilfong. Strictly non-blocking wdm cross-connects for heterogeneous networks. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing (STOC)*, pages 514–523, New York, NY, USA, 2000. ACM.

[16] L. G. Valiant and G. J. Brebner. Universal schemes for parallel communication. In *Proceedings of the thirteenth annual ACM symposium on Theory of computing (STOC)*, 1981.

[17] E. D. Williams. Revisiting energy used to manufacture a desktop computer: hybrid analysis combining process and economic input-output methods. In *Proceedings of the International Symposium on Electronics and the Environment (ISEE '04)*, pages 80–85, Washington, DC, USA, 2004. IEEE Computer Society.

[18] H. Wu, G. Lu, D. Li, C. Guo, and Y. Zhang. MDCube: a high performance network structure for modular data center interconnection. In *CoNEXT '09: Proceedings of the 5th international conference on Emerging networking experiments and technologies*, 2009.

[19] R. Zhang-Shen and N. McKeown. Designing a predictable internet backbone with Valiant load-balancing. In *Thirteenth International Workshop on Quality of Service (IWQoS '05)*, 2005.