Guaranteeing Quality of Service to Peering Traffic

Rui Zhang-Shen
Department of Electrical Engineering
Princeton University
Email: rz@princeton.edu

Nick McKeown
Computer Systems Laboratory
Stanford University
Email: nickm@stanford.edu

***Abstract*—Network operators connect their backbone networks together at peering points. It is well known that the peering points are the most congested parts of the backbone network. Network operators have little incentive to provision them well, and have few tools to decide how best to route traffic over them.**

In this paper we propose how peering networks can be congestion free, so long as we know the total amount of traffic between them. In particular, we propose the use of Valiant Load-Balancing (VLB), which has been previously studied for individual backbone networks. In our approach, the backbone networks do not need to use VLB internally—they simply load-balance traffic over their peering links. Our analysis shows how the load-balancing should be done, and we conclude that no other method is more efficient than VLB in achieving a congestion-free network.

I. Introduction

A. Background

Today, most congestion in backbone networks takes place on the peering links between network operators [5], [1]. This is because peering links tend to be under-provisioned; *i.e.*, the network operators use links that are too small to carry all the traffic during peak periods. It might be surprising that operators do not just increase the capacity of each link — over-provision them — so the network will perform better. After all, each operator's backbone network is heavily over-provisioned – often by an order of magnitude or more [4]. Operators over-provision their backbone networks because of three main types of uncertainty: (1) **Future traffic**. When they deploy a network, it will have to operate for several years, even as an unpredictable number of new customers start to use the network, and as new applications create new traffic patterns; (2) **Failures and re-routing.** When links and routers fail, traffic is routed, and any link might have to carry additional traffic; and (3) **Queueing delay.** Customers do not like queueing delay and will frequently move to a new operator with lower delay and jitter performance. All of these factors provide ample incentive for an operator to over-provision their backbone network — at considerable additional cost — making the network easier to manage, have a longer deployment lifetime, and to keep their customers happy.

So why don't operators over-provision the peering links too? Apparently, they do not have enough of an incentive to do so. If a user's traffic is traversing two networks, and the performance is poor, the customer cannot tell which backbone network is at fault, or if the peering links are congested. Not knowing which network to blame, the user is unlikely to switch providers, and so there is little point in increasing the capacity of the links. Even if the operator wanted to increase the peering links, it is hard to know how large to make them. The size of the links depends not only on the future behavior of their own customers, it also depends on the number, behavior, and location of their peer's customers, which they are unlikely to be able to estimate. If they estimate badly, then some links will be swamped, while other links sit idle.[1]

In summary, it is not clear how a network operator could size their peering links so as to give good performance at a reasonable price, and in the absence of such a method they do not have much incentive to over-provision instead.

In this paper we propose a solution. First, we show a simple mechanism (based on a technique called "Valiant Load-Balancing", or VLB) that allows us to size peering links so as to prevent all congestion, regardless of the particular paths or traffic matrices between two networks. We just need to estimate the total amount of traffic between them. This then leads to a simple evolutionary model, in which new capacity can be added to any peering link and will improve the performance of the whole network. Second, we will show that this mechanism is the most efficient and cost-effective way to prevent congestion in peering networks.

B. Valiant Load-Balancing

In the early 1980s, L.G. Valiant proposed the idea of routing packets through random midpoints for the communication among sparsely connected parallel computers [14], [15]. In recent years, VLB was used to design Internet routers with performance guarantees [2], [3], [6], as well as in achieving high worst-case performance without sacrificing average- and best-case performance in interconnection networks [11]. Several groups independently applied the idea of Valiant Load-Balancing to backbone network design and traffic engineering, to efficiently support all possible traffic matrices [10], [9], [7], [8], [16], [17].

The Homogeneous Case. To illustrate VLB in a backbone network, consider the mesh of long-haul links (represented by the cloud) in Figure 1 that interconnect N backbone nodes. Current backbone networks have about $N = 50$ nodes. The network is hierarchical, and each backbone node connects

[1] Matters are made worse by the common and seemingly cheeky practice of *hot-potato routing* [12], [13], in which network operators push traffic to their peer's network as soon as they can, so as to minimize the load in their own network.

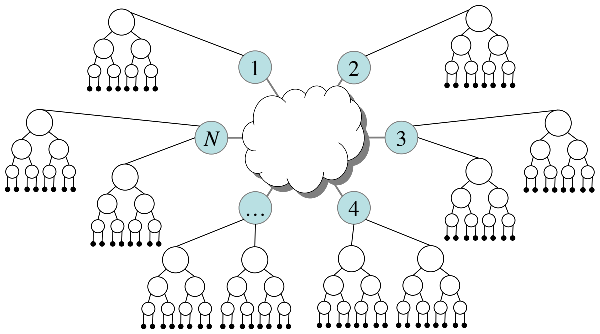

Fig. 1. A hierarchical network with N backbone nodes

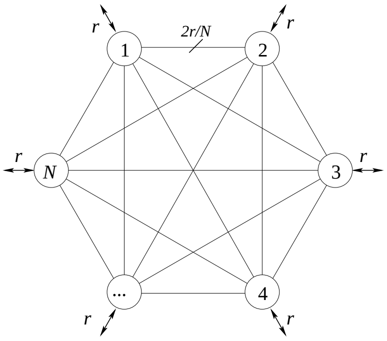

Fig. 2. Valiant Load-Balancing in a network of N identical nodes each having capacity r.

an access network to the backbone. We assume we know (roughly) the total capacity of each access network.

We represent the traffic demand between the backbone nodes by a $N \times N$ traffic matrix, where $\lambda(i,j)$ is the average rate of traffic from node i destined to node j. We say the network can *support* a traffic matrix if the capacity between i and j (either directly or indirectly) is greater than $\lambda(i,j)$.

We will start with the simple (but unrealistic) homogeneous case where all the backbone nodes have the same capacity, r. In this case, a VLB network consists of a full mesh of logical links with capacity $\frac{2r}{N}$, as shown in Figure 2. Traffic entering the backbone is load-balanced equally across all N one- and two-hop paths between ingress and egress. A packet is forwarded twice in the network: In the first hop, a node uniformly load-balances each of its incoming flows to all the N nodes, regardless of the packet destination. Load-balancing can be done packet-by-packet, or flow-by-flow, and each node receives $\frac{1}{N}$ of every flow in the first hop. In the second hop, all packets are delivered to the final destinations.

VLB has the nice characteristic that it can support all traffic matrices that do not oversubscribe a node. Since the incoming traffic rate to each node is at most r, and the traffic is evenly load-balanced to N nodes, the actual traffic on each link due to the first hop routing is at most $\frac{r}{N}$. The second hop is the dual of the first. Since each node can receive traffic at a maximum rate of r and receives $\frac{1}{N}$ of the traffic from every node, the traffic on each link due to the second hop is also at most $\frac{r}{N}$. Therefore, the full-mesh network (with link capacities $\frac{2r}{N}$) can support all traffic matrices. The advantage of VLB for the backbone operator is that they can design their network knowing only the capacities of the access nodes, without knowing anything about the traffic patterns or how they evolve over time. The cost is that the total network has twice the capacity needed, if we knew the actual traffic matrix. It is clear today that backbone operators have little idea what traffic matrices to expect, which explains (in part) why they use five or ten times the minimum capacity. As we have shown elsewhere, VLB networks can be very easily designed to continue working when links and nodes fail, with much lower capacity requirements than existing backbone networks [18].

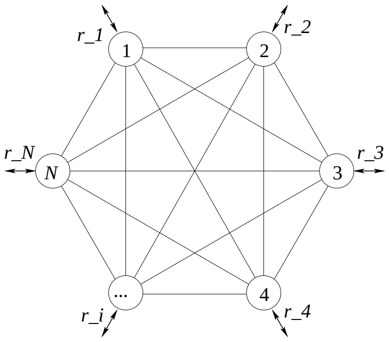

Fig. 3. Valiant Load-Balancing in a heterogeneous N-node network.

The Heterogeneous Case. Of course in practice, the capacity of each access network is different. VLB can be extended quite easily to the heterogeneous case [17]. Uniform load-balancing is no longer the best solution, and it is better to load-balance by sending different amounts of traffic to each node, as a function of the size of the nodes. To illustrate this, consider the N-node network shown in Figure 3. The access capacities of the nodes are r_1, r_2, $\ldots$, r_N, and c_{ij} is the link capacity from node i to node j.[2]

The *interconnection capacity*, l_i, is the total capacity of all the links between node i and other nodes, i.e.,

$$l_i = \sum_{j:j \neq i} c_{ij}. \tag{1}$$

The total capacity of the network, L, is simply

$$L = \sum_{i=1}^{N} l_i = \sum_{i,j:i \neq j} c_{ij}. \tag{2}$$

The maximum amount of traffic that all the access nodes can bring to the network, R, is given by

$$R = \sum_{i=1}^{N} r_i. \tag{3}$$

[2] We assume that a node can send traffic to itself without using any network resource, so we set $c_{ii} = \infty$. Equivalently, we can set the diagonal entries of any given traffic matrix to zero.

The ratio L/R is a measure of the cost of the network. For example, in the homogeneous case, if we know the traffic matrix, we can set $c_{ij} = \lambda_{ij}$ and $L/R = 1$; if we do not know the traffic matrix and must support all of them, then $L/R = 2 - 2/N$ with VLB.

In the heterogeneous case, we will use *oblivious* load-balancing, where flows are split according to the *internal load-balancing parameters* p_i, $i = 1, 2, \ldots, N$, regardless of the flow's source and destination nodes. This scheme has the fewest parameters to set and so is simple to configure in practice. A portion p_i of all flows are load-balanced to intermediate node i in first-hop routing; the intermediate nodes then deliver the traffic to the final destinations in second-hop routing. The *first-hop* traffic on link (i, j) is the traffic from node i that is load-balanced to node j, and is at most $r_i p_j$. The *second-hop* traffic on link (i, j) is the traffic for node j that is load-balanced via node i, and is at most $r_j p_i$. Therefore the maximum amount of traffic on link (i, j) is $r_i p_j + r_j p_i$, so the required capacity on link (i, j) is

$$c_{ij} = r_i p_j + r_j p_i, \tag{4}$$

the interconnection capacity of node i is

$$l_i = \sum_{j:j \neq i} c_{ij} = r_i + R p_i - 2 r_i p_i, \tag{5}$$

and the total interconnection capacity of the network is

$$L = \sum_{i=1}^{N} l_i = \sum_{i,j:i \neq j} c_{ij} = 2 \left(R - \sum_i r_i p_i \right). \tag{6}$$

Equation (6) says that $L/R < 2$ for a heterogeneous network running VLB no matter what the load-balancing parameters are. In [17] we used the notion of *network fanout* to design a VLB scheme that achieves balanced traffic among the nodes, which we summarize here.

We define the *fanout* of node i to be $f_i = l_i/r_i$, the ratio of node i's interconnection capacity to its access capacity. Since the interconnection capacity is used for both first-hop and second-hop traffic, the fanout is at least one, and it measures the amount of responsibility the node has to forward other nodes' traffic relative to its size. If the fanouts of two nodes are the same, then the larger node forwards more traffic, which is a desired property. Our goal is to equalize the fanouts of all the nodes.

Let *network fanout* f be the maximum fanout among all nodes, i.e., $f = \max_i f_i$. Then the network fanout not only shows how efficient the network is but also indicates how balanced the network is, because a smaller network fanout means that the nodes must have similar fanouts.

From [17], the minimum network fanout under oblivious load-balancing is

$$\min f = 1 + \frac{1}{\sum_{j=1}^{N} \frac{r_j}{R - 2r_j}}, \tag{7}$$

with load-balancing parameters

$$p_i = \frac{\frac{r_i}{R-2r_i}}{\sum_k \frac{r_k}{R-2r_k}}. \tag{8}$$

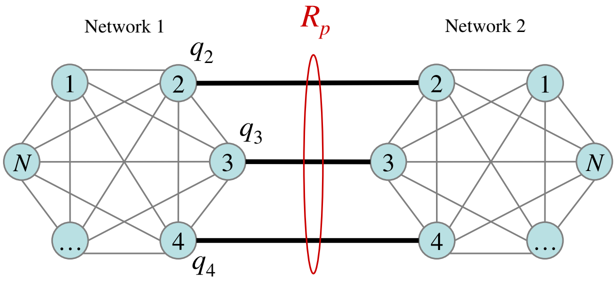

Fig. 4. Two VLB networks connect at a set of peering nodes. The total amount of traffic exchanged between the two networks is no more than R_p, and a portion q_i of the peering traffic is exchanged at node i.

When minimum network fanout is achieved all the nodes have the same fanout, which is equal to the ratio L/R, a number between one and two.

C. Peering Networks

In this paper we show how to interconnect two peering networks so as to minimize congestion between them. In particular, we show that if traffic is load-balanced over all the links between them, there will be no congestion if (and only if) the total peering capacity is greater than the total traffic between them. While we will use VLB between the two networks, they do not need to use VLB internally. But if they do use VLB internally, the performance of the peering traffic will be just as good as the performance of the internal traffic.

Suppose two VLB networks are connected by a subset of their nodes (the peering nodes), as shown in Figure 4. For the ease of description, we use the same numbering on the peering nodes in both networks. The traffic exchanged between the two networks is *peering traffic* and the total amount is no more than R_p in each direction.

We introduce the *peering load-balancing parameters* $q_i, i = 1, 2, \ldots, N$, such that a portion q_i of the peering traffic between the two networks is exchanged at node i. Naturally, $q_i = 0$ if i is not a peering node. Let $\mathcal{P}$ represent the set of peering nodes.

The peering load-balancing parameters together with the maximum peering traffic between the two networks, R_p, determine the size of the peering links: the required capacity of the peering link at node i is $R_p q_i$. Suppose the peering links have the required capacities, then if the peering traffic is load-balanced across the peering links according to the proportions q_i, and the total amount of peering traffic between the two networks does not exceed R_p, there will be no congestion on the peering links.

In what follows, we explore in detail how to use Valiant Load-Balancing to route peering traffic. First, assuming that the peering load-balancing parameters are given, how can the networks provision their internal links to support the peering traffic? Second, what is the most efficient way to split the traffic over all the peering links, and how should this decision

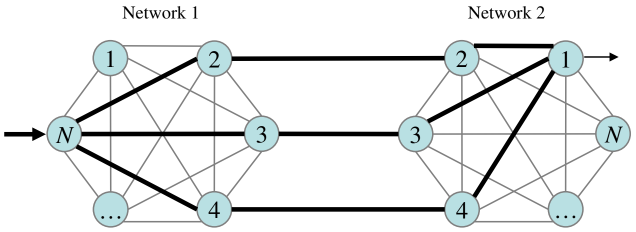

Fig. 5. Delay sensitive load-balanced peering, in which peering traffic is load-balanced over all peering links and traverses exactly one hop in each of the two backbones. The heave lines represent the links that a flow from node N of Network 1 to node 1 of Network 2 traverses.

be made between two competing networks? These questions are explored in Section II and III respectively.

II. Provisioning Internal Links for Peering Traffic

The extra requirement of routing peering traffic may require higher capacity inside the networks. In this section we assume that the peering parameters are fixed, for example, because they are determined by negotiation between the two networks and given in contacts, and consider only one of the two peering networks.

Peering traffic that originates from the network can be treated as traffic destined to the peering nodes and peering traffic that enters the network can be treated as traffic originated from the peering nodes. This is equivalent to increasing the capacity of a peering node by the maximum amount of peering traffic it handles. So if we replace r_i with

$$r'_i = r_i + R_p q_i$$

in Equations (7) and (8), we can find the required link capacity within the network.

At first glance, it would seem that peering traffic has to traverse both networks twice, if they use VLB. This seems inefficient, and could lead to unacceptable latency. Luckily, there is a simple way to avoid this—we can take advantage of the multiple peering points, and load-balance over all the links between the networks. This way, traffic need only traverse each network once (as it does today). Since all Tier 1 ISPs peer with each other, a packet needs to traverse at most two backbones. With one hop in each backbone, peering traffic traverses the network at most twice and has a delay similar to that of the traffic that stays within a VLB backbone. We call this *delay-sensitive load-balanced peering*, which is illustrated in Figure 5.

Now the peering traffic originating in the network must be routed to the peering nodes in one hop; similarly, the peering traffic entering the network must be routed to the destination in one hop. Assuming that non-peering traffic is routed according to the internal load-balancing parameters like before, we will calculate how much capacity is required to route all the traffic.

We use r_i^l to represent the amount of local (non-peering) traffic originated from node i, r_i^p the amount of peering traffic originated from node i, c_i^l the amount of local (non-peering) traffic destined to node i, and c_i^p the amount of peering traffic destined to node i. The quantities are bounded:

$$\begin{aligned} r_i^p &\le \min(r_i, R_p) && (9)\\ r_i^l + r_i^p &\le r_i && (10)\\ c_i^p &\le \min(r_i, R_p) && (11)\\ c_i^l + c_i^p &\le r_i && (12) \end{aligned}$$

Let $t_{ij}^{(1)}$ and $t_{ij}^{(2)}$ represent the first-hop and second-hop traffic from node i to node j, respectively. In first-hop routing, node i sends p_j of its local traffic and q_j of its peering traffic to node j, i.e.,

$$t_{ij}^{(1)} = r_i^l p_j + r_i^p q_j.$$

In second-hop routing, node i delivers p_i of node j's local traffic and q_i of node j's peering traffic, i.e.,

$$t_{ij}^{(2)} = p_i c_j^l + q_i c_j^p.$$

We maximize $t_{ij}^{(1)}$ given the bounds (9-12) and have

$$\max t_{ij}^{(1)} = \begin{cases} r_i p_j, & p_j \ge q_j \\ r_i q_j, & q_j > p_j, R_p \ge r_i \\ r_i p_j + R_p(q_j - p_j), & q_j > p_j, R_p < r_i \end{cases}$$

which can be compactly written as

$$\max t_{ij}^{(1)} = r_i p_j + \min(r_i, R_p)(\max(p_j, q_j) - p_j).$$

Similarly we have

$$\max t_{ij}^{(2)} = r_j p_i + \min(r_j, R_p)(\max(p_i, q_i) - p_i).$$

So the total capacity of link (i, j) is

$$\begin{aligned} c_{ij} &= \max t_{ij}^{(1)} + \max t_{ij}^{(2)} \\ &= r_i p_j + r_j p_i + \min(r_i, R_p)(\max(p_j, q_j) - p_j) \\ &\quad + \min(r_j, R_p)(\max(p_i, q_i) - p_i). \end{aligned}$$

It is the capacity required to support only local traffic, plus some extra terms.

Given the peering load-balancing parameters q_i, we want to find the local load-balancing parameters p_i such that c_{ij} is minimized. We first observe that R_p is likely to be bigger than the node capacities r_i. R_p is the total amount of traffic the two network exchanges and can be a large portion of the network's total traffic R, while the node capacities are likely to make up only a small fraction of R, on the order of $\frac{1}{N}$.

If we assume that $R_p \ge r_i$ for all i, then we have

$$c_{ij} = r_i \max(p_j, q_j) + r_j \max(p_i, q_i),$$

and the minimum c_{ij} is achieved when $p_i = q_i$ for all i. So the optimal capacity allocation in a network with peering traffic is

$$c_{ij} = r_i q_j + r_j q_i. \qquad (13)$$

Since q_i is zero if node i is a non-peering node, $c_{ij} = 0$ if both node i and node j are non-peering nodes. The network is now a two-tiered one: in the center is a full mesh connecting the peering nodes; on the edge are the non-peering nodes, each connecting to all of the peering nodes.

Setting the local load-balancing parameters to be the same as peering load-balancing parameters means that only the peering nodes will serve as intermediate nodes to forward traffic. Peering nodes are often the largest nodes in the network, so they should have larger responsibilities in forwarding traffic. Now the result in this section shows that the optimal way is to only let the peering nodes to forward traffic. This has the additional benefits of requiring fewer links and reducing network complexity.

III. Finding the Optimal Peering Load-Balancing Parameters

In this section we determine the optimal peering load-balancing parameters, namely q_i, for the two networks that peer with each other.

We know from [17] that minimizing the total link capacity in general does not lead to a nice topology. It will likely result in a "star", which has a single point of failure. So we will use the network fanout as our optimization objective because our goal is to design a balanced network.

We will first try to find the optimal peering parameters for one network, assuming that the other network would agree to what this network determines. Then we look at the case of optimizing for both networks at the same time. The formal is useful when the two ASes are cooperative; the latter useful when the two ASes are competing. The objective we use here is to minimize network fanout. However, other objectives can be used according to the design goals.

A. Finding the Optimal Peering Parameters in a Single VLB Network

We continue to assume that R_p is greater than the nodes capacities so that Equation (13) gives the optimal link capacity c_{ij} in a network with peering. So the interconnection capacity of node i is

$$l_i = \sum_{j=1, j\neq i}^{N} c_{ij} = r_i + q_i(R - 2r_i)$$

and the fanout of node i is

$$f_i = \frac{l_i}{r_i} = 1 + q_i \frac{R - 2r_i}{r_i}. \tag{14}$$

Thus, the fanout of a non-peering node is one because $q_i = 0$, and the fanout of a peering nodes is great than one because $q_i > 0$.

We minimize the network fanout:

$$\begin{aligned} \text{minimize} \quad & f = \max_{i=1}^{N} \left(1 + q_i \frac{R - 2r_i}{r_i}\right) \\ \text{subject to} \quad & \sum_{i=1}^{N} q_i = 1 \\ & q_i \geq 0, \quad i = 1, 2, \ldots, N \\ & q_i = 0, \quad i \notin \mathcal{P} \end{aligned}$$

This is equivalent to

$$\begin{aligned} \text{minimize} \quad & f = \max_{i\in\mathcal{P}} \left(1 + q_i \frac{R - 2r_i}{r_i}\right) \\ \text{subject to} \quad & \sum_{i\in\mathcal{P}} q_i = 1 \\ & q_i \geq 0, \quad i \in \mathcal{P} \end{aligned} \tag{15}$$

This optimization can be solved analytically. Rewrite Equation (14) as

$$q_i = (f_i - 1)\frac{r_i}{R - 2r_i}, \tag{16}$$

and we have

$$\sum_{i\in\mathcal{P}} (f_i - 1)\frac{r_i}{R - 2r_i} = \sum_{i\in\mathcal{P}} q_i = 1. \tag{17}$$

So the positive linear combination of f_i is a constant, and in order to minimize $\max_i f_i$ we must have $f_1 = f_2 = \ldots = f_N = f$. Equation (17) becomes

$$(f - 1)\sum_{i\in\mathcal{P}} \frac{r_i}{R - 2r_i} = 1,$$

which in turn gives

$$\min f = 1 + \frac{1}{\sum_{j\in\mathcal{P}} \frac{r_j}{R - 2r_j}}. \tag{18}$$

The load-balancing parameters are:

$$p_i = q_i = \begin{cases} \frac{\frac{r_i}{R-2r_i}}{\sum_{j\in\mathcal{P}} \frac{r_j}{R-2r_j}}, & i \in \mathcal{P} \\ 0, & i \notin \mathcal{P} \end{cases} \tag{19}$$

This is similar to the case without peering traffic except that only the peering nodes have nonzero load-balancing parameters.

The link capacities are

$$c_{ij} = r_i q_j + r_j q_i. \tag{20}$$

B. Finding the Optimal Peering Parameters in Two VLB Networks

Now we optimize across two networks, still assuming that the amount of peering traffic, R_p, is greater than the capacity of any node in both networks. We use superscripts (1) and (2) to differentiate the two networks. Network 1 has N nodes of capacities $r_1^{(1)}, r_2^{(1)}, \ldots, r_N^{(1)}$; Network 2 has M nodes of capacities $r_1^{(2)}, r_2^{(2)}, \ldots, r_M^{(2)}$. Thus, $R^{(1)} = \sum_{i=1}^{N} r_i^{(1)}$ and $R^{(2)} = \sum_{i=1}^{M} r_i^{(2)}$.

The set of peering nodes in Network 1 and Network 2 are $\mathcal{P}^{(1)}$ and $\mathcal{P}^{(2)}$, respectively, and the peering load-balancing parameters are $\{q_i^{(1)}\}$ and $\{q_i^{(2)}\}$, respectively. For the simplicity of notation, we assume that the two nodes connected by a peering link have the same number, so we have $\mathcal{P}^{(1)} = \mathcal{P}^{(2)} = \mathcal{P}$. The peering links are bidirectional, and for efficient use of these links, we assume that the two nodes that directly peer with each other have the same peering traffic load-balancing parameter, i.e., $q_i^{(1)} = q_i^{(2)} = q_i$, $i = 1, 2, \ldots, n$.

In minimizing fanouts, we only need to consider the peering nodes. The fanout of a peering node is given by Equation (14). We minimize the maximum fanout of all the nodes in both networks:

$$\begin{aligned}\text{minimize} \quad & f = \max\left(\max_{i\in\mathcal{P}}\left(1+q_i\frac{R^{(1)}-2r_i^{(1)}}{r_i^{(1)}}\right),\right. \\ & \qquad \left.\max_{i\in\mathcal{P}}\left(1+q_i\frac{R^{(2)}-2r_i^{(2)}}{r_i^{(2)}}\right)\right) \\ \text{subject to} \quad & \sum_{i\in\mathcal{P}} q_i = 1 \\ & q_i \geq 0, \quad i=1,2,\ldots,n\end{aligned}$$

The expression for the maximum fanout in both networks can be simplified as

$$f = \max_{i\in\mathcal{P}}\left(1+q_i\max\left(\frac{R^{(1)}-2r_i^{(1)}}{r_i^{(1)}}, \frac{R^{(2)}-2r_i^{(2)}}{r_i^{(2)}}\right)\right).$$

Compare this to (15) and we can conclude that the optimal solution to the minimum fanout problem is

$$\min f = 1 + \frac{1}{\sum_{i\in\mathcal{P}}\min\left(\frac{r_i^{(1)}}{R^{(1)}-2r_i^{(1)}}, \frac{r_i^{(2)}}{R^{(2)}-2r_i^{(2)}}\right)} \tag{21}$$

and the optimal peering load-balancing parameters are

$$q_i = \begin{cases} \frac{\min\left(\frac{r_i^{(1)}}{R^{(1)}-2r_i^{(1)}}, \frac{r_i^{(2)}}{R^{(2)}-2r_i^{(2)}}\right)}{\sum_{i\in\mathcal{P}}\min\left(\frac{r_i^{(1)}}{R^{(1)}-2r_i^{(1)}}, \frac{r_i^{(2)}}{R^{(2)}-2r_i^{(2)}}\right)}, & i\in\mathcal{P} \\ 0, & i\notin\mathcal{P}\end{cases} \tag{22}$$

C. Discussion

From Equation (7) to Equation (18) to Equation (21), as more conditions are considered in the optimization, the minimum network fanout increases.

When there is only internal traffic, all nodes serve as intermediaries and the fraction of traffic that node i forwards is

$$p_i \propto \frac{r_i}{R-2r_i}. \tag{23}$$

This function increases faster than linear in r_i, so the responsibility of forwarding traffic tends to concentrate in the larger nodes. This is the desired situation because larger nodes usually play more important roles in the network. For example, they may serve a large number of customers or could be gateways to other networks. The load-balancing parameters of the form (23) correspond to a network fanout of less than 2 [17].

Peering traffic usually dominates internal traffic, so when peering traffic is considered, the optimal strategy is to only let the peering nodes forward traffic. This again leads to

$$p_i = q_i \propto \frac{r_i}{R-2r_i},$$

but only for the peering nodes. The load-balancing parameters for non-peering nodes are zero. The fanout of the peering nodes is bigger than without peering traffic because they are forwarding more traffic than before. Now the network fanout depends on the peering nodes and there is no bound on how big it can be, but having more peering nodes leads to a smaller network fanout.

When two networks are jointly optimized, the network fanout only becomes bigger. Now we have

$$p_i = q_i \propto \min\left(\frac{r_i^{(1)}}{R^{(1)}-2r_i^{(1)}}, \frac{r_i^{(2)}}{R^{(2)}-2r_i^{(2)}}\right).$$

For each peering node pair, the node that is a smaller fraction of the total capacity of the network it belongs to dominates the expression. Consider the simple case where Network 1 has N identical nodes and Network 2 has M identical nodes. If Network 1 has more nodes, i.e., $N > M$, then the nodes in Network 1 dominate. So for Network 1, the result of the optimization is as if it were done without considering Network 2. In this simple example the load-balancing parameter are $p_i = q_i = \frac{1}{n}$ if node i is a peering node, where n is the number of peering nodes. This is optimal for both networks and the fanout of Network 2 is even smaller than that of Network 1. But if the nodes are not identical in each network, Network 2 may not achieve its optimal capacity distribution. This means the larger network tends to have an advantage in dominating the optimization for determining the peering load-balancing parameters.

If the two networks are cooperative, then other forms of tradeoffs can be used. For example, if one network has limited resources compared to the other network, then parameters can be determined according to this network instead of by joint optimization.

IV. Conclusions

Valiant Load-Balancing seems like a promising way to design congestion-free backbone networks, but clearly requires quite a radical rethinking by a network operator. A VLB backbone network is essentially over-provisioned by a factor of two; while this is much less over-provisioning than in current networks, we realize this can be a pill emotionally difficult to swallow.

On the other hand, VLB seems like quite a straightforward way to provision and route peering traffic between backbone networks, and requires no over-provisioning between them. We just need to know the total expected load of traffic between the networks, and we do not need to know the specific routes they will take in either network.

If VLB is deployed today, then some thought is needed to route traffic inside each backbone. Current hop-by-hop routing schemes do not support arbitrary load-balancing, so we need tunnels for the VLB peering traffic. MPLS technology provides a flexible way to set up tunnels in the backbone and is used by several Tier 1 ISPs today. So the technical challenge in adopting VLB within a network and between networks is minimal. An alternative way to provide tunnels for VLB is through TDM circuits on the optical fibers. This approach will prove economical in the long run as the datarate on optical fibers increase and IP routers have a harder time to keep up, therefore providing economic incentives to adopt VLB.

V. Acknowledgment

This research was funded by NSF under ITR award ANI-0331653, the Stanford Clean Slate Program, and the Lillie Family Stanford Graduate Fellowship.

We thank John Chuang and Ramesh Johari for our discussions with them on VLB.

References

[1] A. Akella, S. Seshan, and A. Shaikh. An empirical evaluation of wide-area Internet bottlenecks. *SIGMETRICS Perform. Eval. Rev.*, 31(1):316–317, 2003.

[2] C.-S. Chang, D.-S. Lee, and Y.-S. Jou. Load balanced Birkhoff-von Neumann switches, Part I: One-stage buffering. *Computer Communications*, 25(6):611–622, 2002.

[3] C.-S. Chang, D.-S. Lee, and C.-M. Lien. Load balanced Birkhoff-von Neumann switches, part II: Multi-stage buffering. *Computer Communications*, 25(6):623–634, 2002.

[4] C. Fraleigh. *Provisioning IP Backbone Networks to Support Delay Sensitive Traffic*. PhD thesis, Department of Electrical Engineering, Stanford University, 2002.

[5] N. Hu, L. E. Li, Z. M. Mao, P. Steenkiste, and J. Wang. Locating Internet bottlenecks: algorithms, measurements, and implications. In *Proc. ACM SIGCOMM*, pages 41–54, New York, NY, USA, 2004. ACM Press.

[6] I. Keslassy, S.-T. Chuang, K. Yu, D. Miller, M. Horowitz, O. Solgaard, and N. McKeown. Scaling Internet routers using optics. *Proceedings of ACM SIGCOMM, Computer Communication Review*, 33(4):189–200, October 2003.

[7] M. Kodialam, T. V. Lakshman, and S. Sengupta. Efficient and robust routing of highly variable traffic. In *HotNets III*, November 2004.

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

[9] H. Nagesh, V. Poosala, V. Kumar, P. Winzer, and M. Zirngibl. Load-balanced architecture for dynamic traffic. In *Optical Fiber Communication Conference*, March 2005.

[10] F. B. Shepherd and P. J. Winzer. Selective randomized load balancing and mesh networks with changing demands. *Journal of Optical Networking*, 5:320–339, 2006.

[11] A. Singh. *Load-Balanced Routing in Interconnection Networks*. PhD thesis, Department of Electrical Engineering, Stanford University, 2005.

[12] R. Teixeira, A. Shaikh, T. Griffin, and J. Rexford. Dynamics of hot-potato routing in IP networks. *ACM SIGMETRICS Performance Evaluation Review*, 32(1):307–319, 2004.

[13] R. Teixeira, A. Shaikh, T. Griffin, and G. M. Voelker. Network sensitivity to hot-potato disruptions. In *Proc. ACM SIGCOMM*, pages 231–244, New York, NY, USA, 2004. ACM Press.

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

[15] L. G. Valiant and G. J. Brebner. Universal schemes for parallel communication. In *ACM Symposium on Theory of Computing*, pages 263–277, New York, NY, USA, 1981. ACM Press.

[16] R. Zhang-Shen and N. McKeown. Designing a Predictable Internet Backbone Network. In *HotNets III*, November 2004.

[17] R. Zhang-Shen and N. McKeown. Designing a predictable Internet backbone with Valiant Load-Balancing. *Thirteenth International Workshop on Quality of Service (IWQoS)*, 2005.

[18] R. Zhang-Shen and N. McKeown. Designing a Fault-Tolerant Network with Valiant Load-Balancing. In *Proc. IEEE INFOCOM*, April 2008.