

On Direct Routing in the Valiant Load-Balancing Architecture

Huan Liu, Rui Zhang-Shen
 Computer Systems Laboratory
 Stanford University, Stanford, CA 94305-9030
 {huanliu, rzhang}@stanford.edu

Abstract—It is very hard to design a network with performance guarantees, partly because it is hard to estimate the future traffic matrix. With no knowledge of the traffic matrix, one can use the Valiant Load-balancing (VLB) architecture which can support any traffic satisfying the node capacity constraints. To interconnect N nodes of capacity r , the VLB architecture requires a logical full mesh of link capacity $c = \frac{2r}{N}$.

Uniform load-balancing can guarantee throughput, but is not necessary if the traffic matrix is known, in which case the amount of load-balancing can be reduced. In this paper we study adaptive load-balancing in two cases: when only local traffic information is known to a node, and when the network traffic matrix is known. We give linear programming formulations to maximize directly routed traffic in both cases, so as to reduce the average hop count of packets. In the case when the traffic matrix is known, we show that direct routing is not always feasible with $c = \frac{2r}{N}$, but $c = \frac{3r}{N}$ is sufficient.

I. INTRODUCTION

Designing an Internet backbone network that can satisfy the customers' traffic demand is not an easy task. Ideally, if the exact traffic demand matrix is known, the network operators can then provision enough link capacities to carry it. Unfortunately, obtaining the future traffic matrix at design time is not possible, because it is hard to measure the current traffic accurately, and even harder to predict how the traffic will grow. Any prediction will have to involve a lot of guesswork. What's more, even after a network is built, the traffic still fluctuates as usage changes.

Other than the unpredictable traffic, network operators face an even bigger challenge – the components in the network fail, and the network needs to continue carrying traffic when failures happen. This is obviously very hard to do. The only current solution is to grossly over-provision the networks, and typical backbone networks today operate at an average utilization well below 30% [2].

The Valiant Load-balancing (VLB) architecture [6] [7] is able to solve both problems. By using load-balancing, the network is able to support all traffic matrices, and can continue to do so even under failures. This architecture was first proposed by Valiant for processor interconnection networks [5], and has received recent interest for scalable routers with performance guarantees [1] [4] and for designing backbone networks with predictable performance.

The VLB architecture assumes a full mesh of logical links in the backbone, as shown in Fig. 1. In this paper, we consider a backbone network of N identical nodes, each

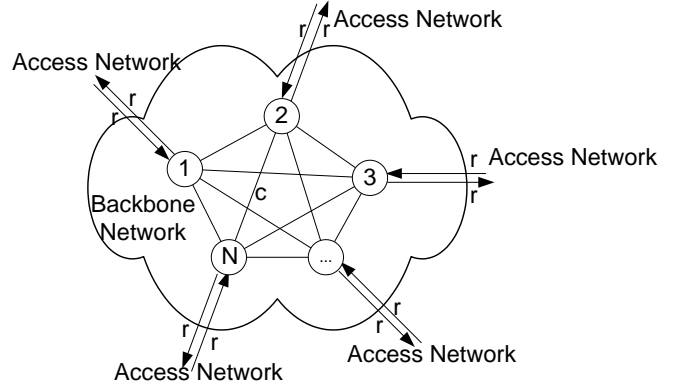


Fig. 1. A Valiant Load-balancing (VLB) backbone network. Each node is connected to an access network that can send and receive traffic at a maximum rate of r . Each pair of the backbone nodes are connected by a link of capacity $c = \frac{2r}{N}$.

serving an access network of capacity r . We use $\Lambda = [\lambda_{ij}]$ to denote the traffic matrix. In order for the traffic matrix to be valid, we require $\sum_i \lambda_{ij} \leq r, \forall j$ and $\sum_j \lambda_{ij} \leq r, \forall i$. In addition, we can assume that the diagonal entries of Λ are zeros. By symmetry, all the links have the same capacity, represented by c . Keslassy et al. [3] proved that the Valiant Load-balancing architecture is the unique fixed interconnect that can support all traffic matrices with the minimum amount of interconnection capacity at each node, and the required link capacity between any pair of nodes is $\frac{2r}{N}$.

The VLB architecture can guarantee 100% throughput for any traffic matrix through uniform load-balancing. A flow (defined by the source and destination nodes) is evenly load-balanced across N two-hop paths from ingress to egress, with each node in the network acting as the intermediary. The routing is done in two stages and each packet traverses the backbone twice: once from the ingress node to an arbitrary intermediate node and once again from the intermediate node to reach the egress node. Due to the N -way even splitting of all flows, the capacity requirement on each link for each routing stage is $\frac{r}{N}$, so a full-mesh of link capacity $\frac{2r}{N}$ can guarantee throughput to any traffic matrix.

One of the advantages of the uniform load-balancing scheme is that the nodes do not need to know the traffic matrix. As long as the traffic matrix does not over-subscribe any node, it can be supported by the network. If we look closer at uniform

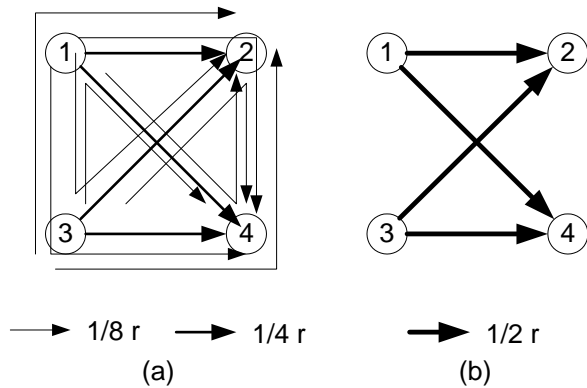


Fig. 2. An example showing the difference between (a) uniform load-balancing and (b) direct routing

load-balancing, we can see that not all traffic traverses two-hop paths. When part of a flow is load-balanced through the source or destination node, the path it takes is actually one-hop, because the other hop is inside a node. So by default $2-N^{\text{th}}$ of each flow takes the direct path from source to destination, and the rest traverses two-hop paths. So the average hop count for all traffic is $2(1 - \frac{1}{N})$.

In this paper, we'd like to explore the following question: if the traffic matrix is known, or partially known during operation, can we send more traffic along the direct paths (and therefore reduce the average hop-count)?

If each link has capacity c , then at most this amount of traffic can be sent directly between two nodes. We use *direct* traffic to refer to the amount of traffic that can be routed directly, i.e., direct traffic from node i to node j is $d_{ij} = \min(\lambda_{ij}, c)$. We use *indirect* traffic to refer to the amount of traffic that cannot be routed directly, i.e., indirect traffic from node i to node j is $e_{ij} = \lambda_{ij} - d_{ij}$.

Given a traffic matrix, *direct routing* is a routing such that all direct traffic is routed directly. Even if the traffic matrix is known, direct routing may not be achievable. So we would like to find an *adaptive* scheme where the nodes adjust the amount of load-balancing according to the traffic, and try to maximize the amount of directly routed traffic.

Note that if more than two hops are allowed for indirect traffic, then direct routing is always achievable (see Appendix). In the rest of the paper, we will limit the path length to two hops.

The benefits of direct routing can be illustrated by a simple example shown in Fig. 2. The network has $N = 4$ nodes and the traffic matrix $\Lambda = [\lambda_{ij}]$ is such that half of the traffic from node 1 and node 3 goes to node 2 and the other half goes to node 4:

$$\Lambda = \begin{bmatrix} 0 & \frac{1}{2}r & 0 & \frac{1}{2}r \\ 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2}r & 0 & \frac{1}{2}r \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Since $N = 4$, the VLB architecture has $c = \frac{2r}{4} = \frac{1}{2}r$ on each link.

To support this traffic matrix, the uniform load-balancing solution will spread the traffic demand between each node pair evenly over all four nodes. Fig. 2(a) is the resulting routing solution, where only half of the traffic is routed directly. Fig. 2(b) is the direct routing solution, where all traffic demands are routed directly. Clearly, direct routing is more desirable compared to uniform load-balancing in this example. The average hop count is one hop in the direct routing solution, compared to 1.5 hops in the uniform load-balancing solution.

By maximizing directly routed traffic, how much improvement can be made in average hop count depends on the actual traffic matrix. When the traffic matrix is a permutation matrix, direct routing is equivalent to uniform load-balancing, therefore there is no improvement. But when the traffic matrix is close to uniform, as is true most of the time, almost all traffic can be routed directly, and the average hop count can be reduced.

The rest of the paper is organized as follows. In Section II, we show the maximum amount of direct routing that can be done if only local traffic information is known to the nodes. In Section III, we study the feasibility of direct routing when the network traffic matrix is known. Section IV concludes the paper.

II. ADAPTIVE LOAD-BALANCING WITH LOCAL TRAFFIC INFORMATION

We will assume in this section that each node has only local traffic information, i.e., it measures the rates of flows originating from it. This assumption requires no dependency on network-wide information. Therefore, the routing scheme does not require synchronization or coordination among the nodes. Each node can implement the adaptive load-balancing scheme independently.

Recall that in uniform load-balancing each link needs a capacity of $\frac{2r}{N}$ because each of the two forwarding stages requires a capacity of $\frac{r}{N}$. Assume the same link capacities and no coordination, then in order to avoid any overload in the network, *each stage cannot use more than $\frac{r}{N}$ of capacity on any link*.

Suppose not, and more than $\frac{r}{N}$ of capacity is used by stage 1 on link (a, b) . Then if node a receives $\frac{r}{N}$ of traffic from the other nodes which it needs to deliver to node b , link (a, b) will be overloaded. If we restrict the amount of traffic node a forwards to node b to $\frac{r}{N} - \epsilon$, then by symmetry it has to be true for all the nodes, and node b will not be able to receive traffic at rate r . This means no more than $\frac{r}{N}$ of capacity should be used by stage 1 on any link. Since it's possible that stage 1 traffic needs to use $\frac{r}{N}$ of capacity on a link, stage 2 traffic should not take more than $\frac{r}{N}$ of capacity on any link either.

To guarantee that stage 2 does not use more than $\frac{r}{N}$ of capacity on any link, we need further restrictions. We use the flow from node a to node b as an example. If node a only knows the rates of flows originating from itself, it should not load-balance flow (a, b) to an intermediate node i ($i \neq b$) more than it does in uniform load-balancing. Without coordination, if all nodes load-balance more traffic to node i , then it may

not be able to deliver all packets to their destinations, i.e., there is overload at node i for the second stage. Node a can, however, load-balance more of flow (a, b) to node b , because in this case the second stage is internal to node b and does not require any network resource.

Therefore, the adaptive load-balancing rule is: *First stage routing cannot use more than half of the capacity on any link, and a node can load-balance more than $1-N^{\text{th}}$ of a flow only to the destination node.*

This rule can be formulated as linear constraints, and we can form a linear programming (LP) problem to, for example, maximize the amount of directly routed traffic from node s . Let x_{ikj} denote the amount of traffic from node i to node j routed via node k . Then $x_{iij} + x_{ijj}$ is the amount of flow (i, j) routed directly. The local traffic information is the rates of flows originating from node s , i.e., $\lambda_{si}, \forall i$. The LP formulation is:

$$\begin{aligned} & \text{maximize} && \sum_{i=1}^N (x_{sii} + x_{ssi}) \\ & \text{subject to} && \sum_{i=1}^N x_{ski} \leq \frac{r}{N}, \quad \forall k \neq s \\ & && x_{sik} \leq \frac{\lambda_{sk}}{N}, \quad \forall i, k, i \neq k \\ & && \sum_{i=1}^N x_{sik} = \lambda_{sk}, \quad \forall k \\ & && x_{ski} \geq 0 \quad \forall k, i \end{aligned}$$

The first two constraints ensure that stage 1 and stage 2 routings use no more than $\frac{r}{N}$ of capacity on any link, respectively.

As an example, let's consider the condition when all traffic from s can be routed directly. We have $x_{sst} \leq \frac{\lambda_{st}}{N}$, and $x_{stt} \leq \frac{r}{N}$, so if the flows originating from node s satisfy

$$\lambda_{st} \leq \frac{r}{N} + \frac{\lambda_{st}}{N}, \forall t \neq s, \quad (1)$$

then all flows originating from node s can be delivered on the direct path. Equation (1) gives $\lambda_{st} \leq \frac{r}{N-1}, \forall t$. This means, if all flows originating from a node have rates no bigger than $\frac{r}{N-1}$, this node can send all the flows along direct paths.

III. MAXIMIZING DIRECT ROUTING GIVEN FULL TRAFFIC MATRIX

In this section we solve the problem of maximizing the amount of directly routed traffic centrally, i.e., we assume that the network traffic matrix is known. With global traffic information, we no longer need to divide the link capacity to two stages, and routing can be formulated as a global optimization problem.

A. The Linear Programming Formulation

Using notations from the previous section, we can formulate an LP to maximize the total directly routed traffic subject to

the link capacity constraints:

$$\begin{aligned} & \text{maximize} && \sum_{i,j=1}^N (x_{iij} + x_{ijj}) \\ & \text{subject to} && \sum_k (x_{ijk} + x_{kij}) \leq \frac{2r}{N} \quad \forall i \neq j \\ & && \sum_k x_{ikj} = \lambda_{ij} \quad \forall i, j \\ & && x_{ijk} \geq 0 \quad \forall i, j, k \end{aligned}$$

Given a traffic matrix at operation time, the solution from the LP will dictate how the traffic will be routed in the network. Also, the LP can be used to verify whether direct routing is feasible. If the objective value from the LP matches the total amount of direct traffic, then direct routing is feasible.

B. A sufficient condition for direct routing with $c = \frac{2r}{N}$

In the following, we first prove a theorem on the amount of residual capacity between a node pair, and then give a sufficient condition for direct routing when $c = \frac{2r}{N}$.

Theorem 1 *Suppose each link has capacity $\frac{2r}{N}$, and we only route the traffic originated from node s or destined to node d . After all traffic except the indirect traffic (e_{sd}) from node s to node d has been routed, there is at least $2e_{sd}$ amount of two-hop capacity left from node s to node d .*

Proof: Let us assume that the indirect traffic from node s to node d is non-zero, because otherwise, the theorem is trivially true. Since there is indirect traffic, the amount of direct traffic from node s to node d is $\frac{2r}{N}$.

The amount of two-hop capacity from s to d is $\frac{2r}{N}(N-2)$. Together with the capacity on the direct link between s and d , the total capacity between the two nodes is $\frac{2r}{N}(N-1)$.

Let f_s denote the total traffic originating from node s other than the indirect traffic to d . This includes the $\frac{2r}{N}$ amount of direct traffic from node s to d . Let f_d denote the total traffic destined to node d other than the indirect traffic from s . This also includes the $\frac{2r}{N}$ amount of direct traffic from node s to d .

Since s can not send more than r amount of traffic and d can not receive more than r amount of traffic, the following should hold:

$$e_{sd} \leq r - \max(f_s, f_d).$$

Let c_{sd}^r denote the capacity left between nodes s and d after all traffic from s and to d other than the indirect traffic between them is routed. Then we have:

$$\begin{aligned} c_{sd}^r &= \frac{2r}{N}(N-1) - f_s - f_d + \frac{2r}{N} \\ &\geq 2r - 2\max(f_s, f_d) \\ &\geq 2e_{sd}. \end{aligned}$$

Note that we add $\frac{2r}{N}$ on the right-hand side because the direct traffic from s to d is double counted in f_s and f_d . ■

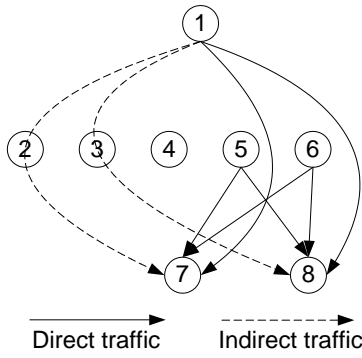


Fig. 3. An example to show that first-hop traffic (from node 1 to 7 and 8) can take up more than half of the residual capacity on some links (links (1,2), (1,3) and (1,4)).

Theorem 2 *If no two flows with nonzero indirect traffic share a start node or an end node, then direct routing is feasible with $c = \frac{2r}{N}$.*

Proof: We first route all the direct traffic. By Theorem 1, the residual two-hop capacity for each indirect flow is at least twice the flow size. We can divide the capacity left on each link into two halves, to serve first and second hop traffic separately. Since the indirect flows do not share start (or end) nodes, they do not share a link on the first (or second) hop. So there is enough capacity in each hop to route all the indirect traffic. ■

The above theorem gives a very strict condition for which direct routing is feasible with $c = \frac{2r}{N}$. If multiple nonzero indirect flows share a start (or end) node, then Theorem 1 is only true for the flow routed the last. That is, before multiple indirect flows sharing a start (or end) node are routed, the residual capacity may not be twice the amount of total indirect traffic.

C. Direct routing not always feasible with $c = \frac{2r}{N}$

The question of whether direct routing is feasible for any traffic matrix can be alternatively stated as: after direct traffic is routed, is it possible to route all indirect traffic using the *residual capacity* – the capacity left after direct traffic is routed – in at most two hops?

If indirect traffic does not share any link in each hop, as in Theorem 2, then direct routing is possible. When this is not true, we can construct an example where direct routing is not possible. We first give an example where multiple indirect flows share first hop capacity (Fig. 3).

Since there are eight nodes, the capacity of each link is $c = \frac{2r}{N} = \frac{r}{4}$. Suppose the traffic matrix is such that node 1 wants to send $\frac{r}{2}$ traffic to node 7 and node 8 respectively; node 5 and 6 each want to send $\frac{r}{4}$ traffic to node 7 and 8 respectively. Node 5 and 6 can send all the traffic directly. Node 1 can send $\frac{r}{4}$ traffic directly to node 7 and node 8 respectively. The remaining traffic has to go through two hops. Since node 5 and 6 both are sending traffic to node 7 and 8, we can only use node 2, 3 and 4 as relay nodes.

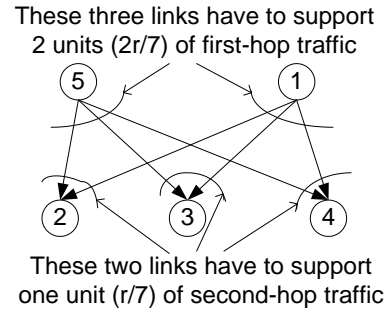


Fig. 4. Routing of the indirect traffic on the selected set of links in the counter-example. There is r amount of indirect traffic to route, but there are only six links each with $\frac{r}{7}$ capacity.

Now we have a total of $\frac{r}{2}$ amount of indirect traffic from node 1 to node 7 and 8, but only $\frac{3r}{4}$ amount of capacity in the first hop on links (1,2), (1,3) and (1,4), less than twice the traffic. As a result, we cannot simply reserve half of the capacity on these links for first-hop traffic and the other half for second-hop traffic.

In the example in Fig. 3, one node sends indirect traffic to two nodes. The indirect traffic for these two pairs share the same first hop capacity. The example can be easily generalized to the case of one node sending indirect traffic to n nodes, where the indirect traffic for these n pairs of nodes needs to share the same first hop capacity. Also, by reversing the flows in the example, we can easily have the case where indirect flows share second-hop capacity.

The example in Fig. 3 points out that we can not simply reserve half of the residual capacity for the first-hop traffic and the other half for the second-hop traffic. We now show the complete counter-example, which uses the example in Fig. 3 as a basic construct, and loads up a set of links with enough first-hop and second-hop traffic such that the sum exceeds the capacity.

The traffic matrix for the counter-example is shown in Table I. The network has 14 nodes, so the capacity on each link is $c = \frac{2r}{N} = \frac{r}{7}$. In the table, we use one marker (x, +, -, *, v) to denote one unit of traffic ($\frac{r}{7}$) and two markers (e.g., ++) to denote two units of traffic ($\frac{2r}{7}$). The counter-example uses the basic construct in Fig. 3 five times. Each use of the basic construct is marked by a different marker. The core of this example is shown in Fig. 4.

The first set of traffic demands, denoted by marker ‘x’, is similar to the example in Fig. 3. Node 1 sends indirect traffic to nodes 5 and 6, but nodes 7 through 14 cannot carry the indirect traffic. As a result, the links (1,2), (1,3) and (1,4) have to support two units of first-hop traffic from node 1. The second set of traffic demands, denoted by marker ‘+’, has indirect traffic from node 5 to nodes 1 and 10, but only nodes 2, 3 and 4 can forward two hop traffic for them. As a result, the links (5,2), (5,3) and (5,4) have to support two units of first-hop traffic from node 5. Note that the amount of first-hop traffic due to these two sets of demands is more than half of the capacity of the set of links ((1,2) (1,3) (1,4) and (5,2) (5,3)

TABLE I
THE TRAFFIC DEMAND MATRIX FOR THE COUNTER-EXAMPLE IN FIG. 4.

from	to													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1					xx	xx	x	x	x					
2			*	v										
3		-		v										
4		-	*											
5	++						+	+	+	++				
6	+	-	*	v									+	
7		--												
8			**				*		*			*	*	*
9				vv			v	v				v	v	v
10		-	*	v	x	x								
11	+	-	*	v	x	x							+	
12	+				x	x							+	
13	+				x	x							+	
14	+				x	x							+	

(5,4).

Having loaded enough first-hop traffic on these links, the next three sets of traffic demands load second-hop traffic on the same set of links. Since the first-hop traffic is already taking up more than half the capacity, it is sufficient to send enough second-hop traffic to take up half of the capacity. To do so, we use a basic construct with only one indirect flow.

The third set of traffic demands, denoted by marker '-', sends indirect traffic from node 7 to node 2 so that links (5,2) and (1,2) need to route one unit of second-hop traffic. The fourth set of traffic demands, denoted by marker '*', sends indirect traffic from node 8 to node 3 so that links (5,3) and (1,3) need to route one unit of second-hop traffic. Lastly, the fifth set of traffic demands, denoted by marker 'v', sends indirect traffic from node 9 to node 4 so that links (5,4) and (1,4) need to route one unit of second-hop traffic.

We now have six links, (1,2) (1,3) (1,4) (5,2) (5,3) and (5,4), each with one unit of capacity, which have to route three units of second-hop traffic and four units of first-hop traffic (see Fig. 4). This is not possible, which means direct routing is not feasible in this network.

In this counter-example, the total direct traffic is $\frac{63r}{7}$, but the LP gives a maximum direct traffic of $\frac{62r}{7}$.

D. Sufficient link capacity for direct routing

We have shown a counter-example where link capacity $c = \frac{2r}{N}$ is not sufficient to guarantee that direct routing is always feasible. In this subsection we will find the sufficient link capacity.

The reason that $c = \frac{2r}{N}$ is not sufficient is because traffic from one stage could use more than half the residual capacity on available links, e.g., first-hop traffic on links (1, 2), (1, 3) and (1, 4) in Fig. 3. If we can make sure that after direct traffic is routed, neither the first-hop traffic nor the second-hop traffic will exceed half the residual capacity on each link, then we

can guarantee that all indirect traffic can be routed. Since the first and second hop routings are symmetric of each other, we only need to make sure that the first-hop traffic does not exceed half of the residual capacity.

To derive a sufficient condition, we consider the case where one node (node s) sends indirect traffic to n other nodes. Suppose node s has the largest amount of indirect traffic to node t , and let δ denote the indirect traffic from node s to node t . We consider the largest indirect flow because this gives us a tighter bound.

The total capacity coming out of node s is $(N - 1)c$. Out of this, at least nc is used to route direct traffic from node s to the n destination nodes. Node t receives δ indirect traffic from node s , therefore, at most $r - \delta$ capacity is taken by direct traffic and cannot be used to route the indirect traffic from node s . Since we are double counting the direct traffic between the node pair (s, t) , the amount of residual capacity that can be used to route indirect traffic is $(N - 1)c - nc - (r - \delta) + c$. This residual capacity could be shared by all indirect flows originating from node s in the first hop.

The source node has at most $r - nc$ indirect traffic that it can send to the n destination nodes through two-hops. We want the residual capacity on the first hop to be at least twice this amount. Therefore, the following condition has to hold:

$$Nc - nc - (r - \delta) \geq 2(r - nc),$$

or

$$3r - (N + n)c \leq \delta. \quad (2)$$

Since the source node can send as much as $r - nc$ indirect traffic, in the worst case δ is at least $\frac{r - nc}{n}$. Substituting δ with $\frac{r - nc}{n}$ in the above equation, we have

$$c \geq \frac{3 - \frac{1}{n}}{n - 1 + N}r. \quad (3)$$

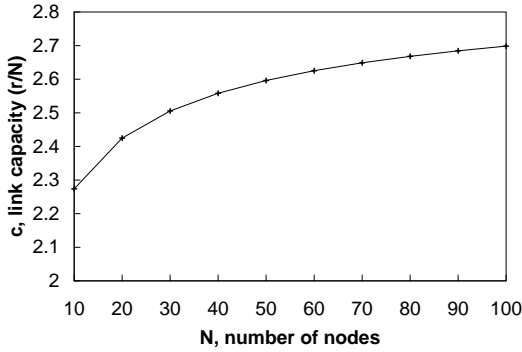


Fig. 5. The link capacity c sufficient to guarantee direct routing vs. N . Link capacity c approaches $\frac{3r}{N}$ from below when $N \rightarrow \infty$.

We can differentiate the above equation to obtain the n that maximizes the lower bound of c :

$$n = \frac{1 + \sqrt{3N - 2}}{3}. \quad (4)$$

We plug this back into (3), then in the limit when $N \rightarrow \infty$, $c/\frac{r}{N} \rightarrow 3$ from below.

In Fig. 5, we plot $c/\frac{r}{N}$ versus N . For practical size networks ($N \leq 100$), less than 35% extra capacity (compared to $c = \frac{2r}{N}$) is required to guarantee that direct routing is feasible.

IV. CONCLUSION

The Valiant Load-balancing architecture is a promising alternative in network design, as it can support any traffic matrix with uniform load-balancing, without requiring any knowledge of the traffic matrix. Naturally, uniform load-balancing is not always necessary if the traffic matrix is known at operation time. In this paper we give LP formulations to maximize the amount of directly routed traffic, given partial or complete traffic matrix. We also showed that direct routing is not always feasible when the link capacity is $c = \frac{2r}{N}$, but it is feasible when $c = \frac{3r}{N}$.

REFERENCES

- [1] C.-S. Chang, D.-S. Lee, Y.-S. Jou, "Load balanced Birkhoff-von Neumann switches, part I: one-stage buffering," *IEEE HPSR '01*, Dallas, May 2001.
- [2] S. Iyer, S. Bhattacharyya, N. Taft, N. McKeown, C. Diot, "An approach to alleviate link overload as observed on an IP backbone," in *IEEE Infocom*, San Francisco, March 2003.
- [3] I. Keslassy, C.-S. Chang, N. McKeown, D.-S. Lee, "Optimal Load-Balancing", *Proc. Infocom*, Miami, Florida, March 2005.
- [4] I. Keslassy, S.-T. Chuang, K. Yu, D. Miller, M. Horowitz, O. Solgaard, N. McKeown, "Scaling Internet Routers Using Optics," *Proc. of ACM SIGCOMM '03*, Karlsruhe, Germany, August 2003. Also in *Computer Communication Review*, Vol. 33, No. 4, pp. 189-200, October 2003.
- [5] L. G. Valiant, "A scheme for fast parallel communication," *SIAM Journal on Computing*, Vol. 11, No. 2, pp. 350-361, 1982.
- [6] R. Zhang-Shen, N. McKeown, "Designing a predictable Internet backbone network," In *Proc. HotNet III*, Nov. 2004.

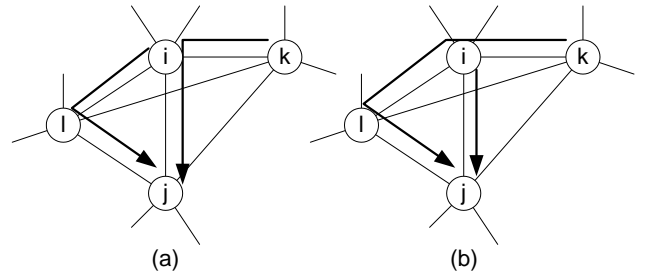


Fig. 6. A procedure to transfer direct traffic (from node i to node j) to direct path, as part of transforming uniform load-balancing to direct routing.

- [7] R. Zhang-Shen, N. McKeown, "Designing a Predictable Internet Backbone with Valiant Load-Balancing," in *Thirteenth International Workshop on Quality of Service (IWQoS)*, Passau, Germany, June 2005.

APPENDIX

Direct Routing with Multi-hop Load-balancing

In this appendix we will show that, if multiple hops can be used to route indirect traffic, we can always achieve *direct routing*. Since the uniform load-balancing solution can always be found as proved in [3], we only need to show that we can transform the uniform load-balancing solution to a direct routing solution with multiple hops.

We show the transformation procedure using the example shown in Fig. 6. Part of the uniform load-balancing solution is shown in Fig. 6(a). Suppose some direct traffic from node i to node j is not routed directly. Then there is either capacity left on link (i, j) or node i is relaying traffic for other nodes to node j . We first move as much as possible direct traffic to link (i, j) to use up any capacity left on the link. If there is still direct traffic from node i to node j that's not routed directly, node i must be relaying traffic for some other node (say node k). We can now swap the routes for the two partial flows as shown in Fig. 6(b). Since the two flows may not be of the same size, the amount of traffic swapped is the minimum of the two flows. This procedure can be repeated as long as there is some direct traffic not routed directly. Note that after the swapping shown in Fig. 6, flow $k-i-j$ travels one more hop than before.

The procedure will terminate after a finite number of steps, because each link carries a finite number of flows, so in order to move the direct traffic to the direct link, the number of swaps required is finite; and there are a finite number of links.

At the end of the procedure, some traffic could be routed through many hops. There is no known bound on the number of hops some traffic may take, other than the obvious bound $N - 1$. Also, the average hop count is not necessarily reduced by this procedure.