Minimizing Average Latency in Oblivious Routing

Prahladh Harsha[*] Thomas P. Hayes[*] Hariharan Narayanan[†] Harald Räcke[‡]
Jaikumar Radhakrishnan[§]

Abstract

We consider the problem of minimizing average latency cost while obliviously routing traffic in a network with linear latency functions. This is roughly equivalent to minimizing the function $\sum_e (\text{load}(e))^2$, where for a network link e, load(e) denotes the amount of traffic that has to be forwarded by the link.

We show that for the case when all routing requests are directed to a single target, there is a routing scheme with competitive ratio $O(\log n)$, where n denotes the number of nodes in the network. As a lower bound we show that no oblivious scheme can obtain a competitive ratio of better than $\Omega(\sqrt{\log n})$.

This latter result gives a qualitative difference in the performance that can be achieved by oblivious algorithms and by adaptive online algorithms, respectively, since there exist a constant competitive online routing algorithm for the cost-measure of average latency [2]. Such a qualitative difference (in general undirected networks) between the performance of online algorithms and oblivious algorithms was not known for other cost measures (e.g. edge-congestion).

1 Introduction

Oblivious routing deals with the design of routing protocols that do not dynamically adapt to the particular traffic pattern (e.g., in a parallel application), but instead use static precomputed routing tables for selecting routing paths. The routing path (or paths in the case of splittable traffic) used between two nodes s and t only depends on the source s the target t (and possibly on some random input in the case of randomized algorithms). Because of their static nature oblivious routing protocols can be implemented very efficiently in a distributed environment, which makes them very attractive from a practical point of view.

However, an important question in this area is whether obliviousness is too simplistic an approach to guarantee good routing performance, and whether one has to resort to adaptive protocols instead. For undirected networks it has been shown that oblivious algorithms perform remarkably well for several cost-functions.

Work in this area was initiated by Valiant and Brebner [11] who developed an oblivious routing protocol for routing in the hypercube that routes any permutation in time that is only a logarithmic factor away from optimal. For the cost-measure congestion (the maximum load of a network link) in a virtual circuit routing model, Räcke [9] proved the existence of an oblivious routing scheme with polylogarithmic competitive ratio for any undirected network. This result was subsequently made constructive by Harrelson, Hildrum and Rao [8] and improved to a competitive ratio of $O(\log^2 n \log\log n)$.

Gupta et al. present oblivious algorithms for a fairly general set of cost-functions, including functions of the form $\sum_e \ell(\text{traffic along } e)$, where ℓ is a concave cost-function. They obtain a competitive ratio of $O(\log^2 n)$ when the demands have to be routed along single paths (integral setting), and a competitive ratio of $O(\log n)$ if demands can be split among paths (fractional setting), and thus can be routed using a flow from the source to the target vertex.

In this paper we study oblivious routing algorithms for the cost-measure of average latency. Assume that we are given linear latency function on the links of the network, i.e., the latency of a network link e that has to forward traffic $f(e)$ is $r_e \cdot f(e)$ for some parameter r_e depending on the network link. The latency of a path is the sum of latencies of all its edges. The total latency of all connections is then equal to $\sum_e r_e \cdot (f(e))^2$.

Since the function $(f(e))^2$ is convex, this important cost-measure of total latency (or equivalently average latency) is not covered by the work of Gupta et al. [5]. In this work we make an initial approach to handle this cost-measure by investigating a restricted version of the problem where all routing requests are directed to the same target node in the network. We analyze

[*]Toyota Technological Institute, Chicago, USA. Emails: {prahladh,hayest}@tti-c.org.

[†]Department of Computer Science, University of Chicago, USA. Email: hari@cs.uchicago.edu.

[‡]Department of Computer Science, University of Warwick, Coventry, UK. Email: H.Raecke@warwick.ac.uk. The author acknowledges the support of DIMAP (the Centre for Discrete mathematics and its Applications) during this work.

[§]Tata Institute of Fundamental Research, Mumbai, INDIA. Email: jaikumar@tifr.res.in.

what we call the *naïve* oblivious routing algorithm in a fractional setting: A source node, upon receiving a request of demand d, routes this demand according to a flow that would be optimal if no other source node were active. It turns out that this simple algorithm obtains a logarithmic competitive ratio:

MAIN THEOREM. *Consider a network of n nodes with linear latency functions on the edges. The naïve oblivious routing algorithm obtains a competitive ratio of $O(\log n)$ with respect to minimizing the average latency when routing to a single target.*

We further show a matching lower bound on the performance of the naïve algorithm and a lower bound of $O(\sqrt{\log n})$ on the competitive ratio of any oblivious routing scheme

Proof Techniques. We obtain both our upper and lower bounds by exploiting the well-known analogy (cf., the excellent monograph of Doyle and Snell [4]) between (a) single target routing networks with quadratic cost function and (b) electrical resistive networks (see Section 3 for more details on this analogy). By this analogy, a flow in the routing network corresponds to a current in the analogous resistive network while the routing cost corresponds to the energy dissipated by the resistive network. When viewed in this manner, the optimal routing corresponds to the current distribution with the minimum energy dissipation (where source node s_i pumps D_i current to the common target). By properties of electrical networks, this optimal current distribution is the super-imposition (with cancellation) of the optimal current distributions obtained by having each of the source nodes active individually (i.e., source node s_i pumps D_i current while all other source nodes pump zero current). We now observe that the naïve oblivious routing algorithm routes exactly according to these individual current distributions, in other words, according to the optimal current distributions obtained by having each source node active individually. Thus the only difference between the optimal routing and the naïve oblivious routing is that the optimal super-imposes these current distributions allowing for cancellation while the oblivious routing super-imposes them *without* cancellation.

The main contribution of the paper is to exploit the above relationship by deriving bounds on the ratio between the costs of these two super-impositions (*with* and *without* cancellation). In Section 3 we show that the ratio is bounded by $O(\log n)$ which gives our Main Theorem. Subsequently, in Section 4 we show that the ratio can become as large as $\Omega(\log n)$ which gives a lower bound on the performance of the naïve oblivious routing scheme. Finally in Section 5 we prove that the competitive ratio of every oblivious algorithms is $\Omega(\sqrt{\log n})$ even for the special case when all sources route to a single target.

1.1 Related Work.

The main body of work about oblivious routing (see e.g. [9, 7, 6, 1, 3]) aims at minimizing the edge congestion in the network. Also the initial work by Valiant and Brebner [11] that uses permutation time as cost-measure is based on a path-selection mechanism that produces low edge-congestion.

As mentioned earlier the cost-measure average latency in a network with linear latency functions is essentially equivalent to minimizing the $\|\cdot\|_2^2$-norm of the link-loads (at least when all resistances r_e are 1). An online algorithm with constant competitive ratio for this problem follows from the work of Awerbuch et al. [2]. This means there is an adaptive routing algorithm with competitive ratio $O(1)$, while our lower bound in Section 4 shows that any oblivious algorithm has a competitive ratio of $\Omega(\sqrt{\log n})$. This means that for the cost-measure average latency there exists a qualitative difference between the performance that can be obtained by oblivious algorithms on the one hand and adaptive algorithms on the other. This is not true for the cost-measure of edge-congestion where it is still possible that in undirected graphs the performance of oblivious algorithms and of adaptive algorithms are asymptotically equal.

The model of using linear latency functions and optimizing for the average latency has been used quite frequently in the literature, most notably in the area of game theory. For example Roughgarden and Tardos [10] analyze routing problems in a game theoretic setting where the path-selection is performed by selfish agents. They study the cost-measure average latency in networks with linear latency function on the edges. However, in contrast to our model their latency functions are allowed to have constant offsets while we require that the latency of a link is directly proportional to its load.

2 Definitions, problem statement, results.

We represent the network as an undirected graph $G = (V, E)$, where V denotes the set of vertices (or nodes) and E the set of edges. Each edge $e \in E$ has an associated *resistance* r_e, that models the latency behavior of the edge e as described below. There is a distinguished target node t. We use n to refer to the number of vertices in G, and often identify V with the set $[n] = \{1, 2 \dots, n\}$. Though the network is undirected and links are allowed to carry traffic in both directions simultaneously, it will be convenient to give each edge an orientation. For an edge e of the form $\{v, w\}$, we will write $e = (v, w)$ when we want to emphasize that the edge is oriented from v to w. The traffic on edge e will be a real number. If this number is positive, it will represent traffic along e from v to w; if it is negative, it will represent traffic along e from w to v. Let $\mathrm{In}(v)$ be the edges of G that are oriented into v and $\mathrm{Out}(v)$ be the edges of G that are oriented away from v.

Definition 2.1. (Flow, costs, circulation) *A flow in G from i to t with value $d \geq 0$ is an assignment $\vec{f}: E \to \mathbb{R}$ such that, for all $v \in V(G) \setminus \{i, t\}$,*

$$\sum_{e \in \mathrm{In}(v)} \vec{f}[e] - \sum_{e \in \mathrm{Out}(v)} \vec{f}[e] = 0, \tag{2.1}$$

and

$$\sum_{e \in \mathrm{In}(t)} \vec{f}[e] - \sum_{e \in \mathrm{Out}(t)} \vec{f}[e] = d;$$

$$\sum_{e \in \mathrm{Out}(i)} \vec{f}[e] - \sum_{e \in \mathrm{In}(i)} \vec{f}[e] = d.$$

We will need the following generalization of this terminology for describing simultaneous flows from multiple sources. We say that a sequence of flows $\vec{f} = \langle \vec{f_i} : i \in V(G) \rangle$ meets the demand $\langle D_i : i \in V \rangle$, if for all $i \in V$, $\vec{f_i}$ is a flow from i to t of value D_i. The load on the edge e under f is given by

$$\ell^f[e] = \sum_i |\vec{f_i}[e]|.$$

The cost incurred on account of this load is $\mathrm{r}_e \cdot (\ell^f[e])^2$. Thus the total cost of $\vec{f}$ is $\mathrm{cost}(\vec{f}) = \sum_e \mathrm{r}_e \cdot (\ell^f[e])^2 = \sum_{ij} \sum_e \mathrm{r}_e \cdot |\vec{f_i}[e] \cdot \vec{f_j}[e]|$.

A circulation in G is an assignment $\vec{c}: E \to \mathbb{R}$, such that for all vertices $v \in V(G)$,

$$\sum_{e \in \mathrm{In}(v)} \vec{c}[e] - \sum_{e \in \mathrm{Out}(v)} \vec{c}[e] = 0. \tag{2.2}$$

Problem statement: Given a instance $\langle G, R, D \rangle$ where $G = (V, E)$ is a graph with associated resistances $R = \langle \mathrm{r}_e : e \in E \rangle$ and demands $D = \langle D_i : i \in V \rangle$, determine flows $\vec{f} = \langle \vec{f_i} : i \in V \rangle$ such that $\vec{f}$ meets D and has minimal cost.

Nature of the optimal solution: The minimum cost solution to this problem can be obtained by viewing G as an electrical network. Let $\vec{J} = \langle \vec{J}[e] : e \in E \rangle$ be the currents that arise in the edges of G, when a current of D_i is injected into vertex i, and a current of $\sum_i D_i$ is drawn out of t. These currents in $\vec{J}$ can be expressed as the sum of flows $\vec{o} = \langle \vec{o}_i : i \in V \rangle$, such that for each edge $e \in E$, all $\vec{o}_i[e]$ have the same sign (independent of i), and $\vec{o}$ meets the demands in D. We will use the following well-known facts from electrical networks.

Fact 2.2. (Thompson's principle) *The solution $\vec{o}$ obtained using the currents in $\vec{J}$ has the minimum cost.*

The currents in $\vec{J}$ can themselves be obtained by combining the optimal solution for each source, considered in isolation. Let $\vec{f_i}$ be the optimal solution when the i-th source has demand D_i, other sources have zero demands. By Fact 2.2, $\langle \vec{f_i}[e] : e \in E \rangle$ can be interpreted as currents that arise when a current of D_i is injected into i and a current of D_i is drawn out if t. This current satisfies Kirchhoff's laws, In particular, there exist potentials $\langle \varphi[i] : i \in V \rangle$ such that for edge $e \in E$ of the form (v, w),

$$\mathrm{r}_e \cdot \vec{f_i}[e] = \varphi[v] - \varphi[w]. \tag{2.3}$$

Fact 2.3. (Superposition principle) *The currents $\langle \vec{J}[e] : e \in E \rangle$ are given by $\vec{J}[e] = \sum_i \vec{f_i}[e]$.*

It follows that the optimum cost for the problem is

$$\begin{aligned} \mathrm{Opt}(G, R, D) &= \sum_e \mathrm{r}_e \cdot \left(\sum_i \vec{f_i}[e] \right)^2 \\ &= \sum_{ij} \sum_e \mathrm{r}_e \cdot \vec{f_i}[e] \cdot \vec{f_j}[e] \ . \end{aligned} \tag{2.4}$$

Oblivious algorithms: Note that the optimal solution obtained by viewing G as an electrical network requires knowledge of the entire demand vector. An oblivious algorithm determines the flow from source i to t, based only on D_i. More precisely, an oblivious algorithm specifies, for each source i (based on the input graph G, and the resistances $\langle \mathrm{r}_e : e \in E \rangle$), a flow $\vec{f}_i^1$ from i to t of unit value. Then given an arbitrary demand vector D, the solution provided by the algorithm is $\langle D_i \cdot \vec{f}_i^1 : i \in V \rangle$, that is it scales $\vec{f}_i^1$ by factor D_i and puts them all together. The goal of this paper is to compare the costs of solutions provided by oblivious algorithms and the optimum cost.

The naïve oblivious algorithm: Facts 2.2 and 2.3, suggest the following simple oblivious algorithm. Let $\vec{f}_i^1$ be the minimum-cost flow from i to t of unit value. Thus for the demand vector D, the solution produced by this algorithm is $\vec{f} = \langle D_i \cdot \vec{f}_i^1 : i \in V \rangle$. Note that $D_i \cdot \vec{f}_i^1$ is the minimum cost flow from i to t of value D_i. Thus each source routes its demand optimally through the network assuming naïvely that the other demands don't exist. The cost of this naïve algorithm is then

$$\text{NAÏVE}(G,R,D)=\sum_{ij}\sum_{e}\mathrm{r}_e\cdot|\vec{f}_i[e]\cdot\vec{f}_j[e]|, \tag{2.5}$$

where $\vec{f}_i = D_i\vec{f}_i^1$ is the minimum cost flow in G from i to t with value D_i. The Main Theorem can now be restated as follows:

THEOREM 2.4. *For all graphs G on n vertices and and all demand vectors D,* $\text{NAÏVE}(G,R,D) = O(\log n)\cdot \text{OPT}(G,R,D)$.

This theorem is proven in Section 3. In Section 4 we provide lower bounds. We first show that our analysis of the naïve oblivious routing algorithm is tight by presenting a network and a demand-distribution for which this algorithm exhibits a cost that is an $\Omega(\log n)$-factor larger than the optimum possible cost. Then, in Section 5, we also provide a lower bound of $\Omega(\sqrt{\log n})$ on the competitive ratio of any oblivious routing algorithm in our cost model.

3 Naïve isn't that bad

We want to show that the ratio between NAÏVE and OPT is small. A natural way would be to use (2.5) and (2.4) show that the ratio is small term-by-term, that is, for all pairs (f_i, f_j), the ratio between $\sum_e \mathrm{r}_e\cdot\vec{f}_i[e]\cdot\vec{f}_j[e]$ and $\sum_e \mathrm{r}_e\cdot|\vec{f}_i[e]\cdot\vec{f}_j[e]|$ is small, where $\vec{f}_i$ is the optimum routing for the i-th demand D_i. This, unfortunately, is not true. However, we will describe a slight modification that we can prove, and which is sufficient to establish Theorem 2.4. First, we need some notation. For a pair of flows $(\vec{f},\vec{g})$, let

$$\begin{aligned}
E^+_{fg} &= \{e : \vec{f}[e]\cdot\vec{g}[e]\ge 0\};\\
E^-_{fg} &= \{e : \vec{f}[e]\cdot\vec{g}[e] < 0\};\\
X^+_{fg} &= \sum_{e\in E^+_{fg}} \mathrm{r}_e\cdot\vec{f}[e]\cdot\vec{g}[e];\\
X^-_{fg} &= \sum_{e\in E^-_{fg}} \mathrm{r}_e\cdot|\vec{f}[e]\cdot\vec{g}[e]|.
\end{aligned}$$

When the pair of flows is $(\vec{f}_i,\vec{f}_j)$, we write E^+_{ij}, E^-_{ij}, X^+_{ij} and X^-_{ij} instead of $E^+_{f_if_j}$, $E^-_{f_if_j}$, $X^+_{f_if_j}$ and $X^-_{f_if_j}$. Let

$$X^+ = \sum_{ij} X^+_{ij} \quad\text{and}\quad X^- = \sum_{ij} X^-_{ij}.$$

Observe that $\text{OPT} = X^+ - X^-$ while $\text{NAÏVE} = X^+ + X^-$. Given this, ideally we would like to show that $X^+_{ij} + X^-_{ij} \le C\cdot(X^+_{ij} - X^-_{ij})$, for some factor $C = O(\log n)$, for then we would get our result by just summing over all pairs i, j. This is equivalent to showing that $X^-_{ij} \le (\frac{C-1}{2})\cdot(X^+_{ij} - X^-_{ij})$. We will instead show the following slightly weaker bound, which will suffice for our purposes.

LEMMA 3.1. *Let t be a "target" vertex in G, and suppose $\vec{f}$ and $\vec{g}$ are flows in G, each satisfying Kirchhoff's laws, and each having a unique sink at t. Then*

$$X^-_{fg} = O(\log n)(X^+_{fg} - X^-_{fg}) + \frac{1}{n}\left(\text{cost}(\vec{f}) + \text{cost}(\vec{g})\right).$$

Before proceeding to prove this lemma (which will take a while), let use see how it implies the theorem. We have

$$\text{NAÏVE} \le X^+ + X^- \le (X^+ - X^-) + 2X^-.$$

The first term on the RHS is OPT. We now use Lemma 3.1 to bound the second term by $O(\log n)\cdot\text{OPT}$. Indeed, by invoking Lemma 3.1 with the pair of flows $(\vec{f}_i,\vec{f}_j)$, we obtain

$$X^-_{ij} = O(\log n)(X^+_{ij} - X^-_{ij}) + \frac{1}{n}\left(\text{cost}(\vec{f}_i) + \text{cost}(\vec{f}_j)\right).$$

By summing this over all pairs (i,j), we obtain

$$\begin{aligned}
X^- &= O(\log n)(X^+ - X^-)\\
&\quad + \frac{1}{n}\sum_{ij}\left(\text{cost}(\vec{f}_i) + \text{cost}(\vec{f}_j)\right)\\
&= O(\log n)\cdot\text{OPT} + 2\sum_i \text{cost}(\vec{f}_i).
\end{aligned}$$

Since $\vec{f}_i$ is the optimum routing for the i-th demand (ignoring other demands), the cost incurred for routing the i-th demand in the optimal solution must be at least $\text{cost}(\vec{f}_i)$. Thus $\sum_i \text{cost}(\vec{f}_i) \le \text{OPT}$. This completes the proof of Theorem 2.4 assuming Lemma 3.1.

Proof of Lemma 3.1. For a pair of flows $(\vec{f},\vec{g})$, and for $\ell = 0, \pm1, \pm2, \ldots$, let

$$\begin{aligned}
E^-_{fg}(\ell) &= \left\{e\in E^-_{fg} : \frac{4^\ell}{2} \le \frac{|\vec{f}[e]|}{|\vec{g}[e]|} < 2\cdot 4^\ell\right\};\\
X^-_{fg}(\ell) &= \sum_{e\in E^-_{fg}(\ell)} \mathrm{r}_e\cdot|\vec{f}[e]\cdot\vec{g}[e]|.
\end{aligned}$$

Using this notation, we have

$$X^-_{fg} = \sum_{\ell=-\lceil\log_4 n\rceil}^{\lceil\log_4 n\rceil} X^-_{fg}(\ell) + \sum_{|\ell|>\lceil\log_4 n\rceil} X^-_{fg}(\ell).$$

Lemma 3.1 will follow immediately if we show that for all (f,g)

$$X^-_{fg}(\ell) \le 2(X^+_{fg} - X^-_{fg}); \tag{3.6}$$

$$\sum_{|\ell|>\lceil\log_4 n\rceil} X^-_{fg}(\ell) \le \frac{1}{n}\left(\text{cost}(f) + \text{cost}(g)\right). \tag{3.7}$$

The proof of (3.7) is simpler, so let us get it out of the way.

Proof of (3.7). For $e \in E^-_{fg}(\ell)$ for $|\ell| > \lceil \log_4 n \rceil$, we have $\max\{|\vec{f}[e]|, |\vec{g}[e]|\} \geq n \cdot \min\{|\vec{f}[e]|, |\vec{g}[e]|\}$. Thus $|\vec{f}[e] \cdot \vec{g}[e]| \leq \frac{1}{n} \cdot (\vec{f}^2[e] + \vec{g}^2[e])$. It follows that

$$\begin{aligned}\sum_{|\ell|>\lceil \log_4 n\rceil} X^-_{fg}(\ell) &= \sum_{|\ell|>\lceil \log_4 n\rceil} \sum_{e\in E^-_{fg}(\ell)} \mathrm{r}_e \cdot |\vec{f}[e] \cdot \vec{g}[e]| \\ &\leq \sum_{e\in E} \mathrm{r}_e \cdot \frac{(f^2[e] + g^2[e])}{n} \\ &= \frac{1}{n}(\mathrm{cost}(f) + \mathrm{cost}(g)).\end{aligned}$$

Proof of (3.6). We need to show (3.6) only for $\ell = 0$, because the claim for other ℓ follows from this special case by scaling f by a factor 4^ℓ. To see this, assume that (3.6) holds when $\ell = 0$, for all choices (f, g) satisfying the assumption of Lemma 3.1. Then

$$\begin{aligned}X^-_{fg}(\ell) &= 4^{-\ell} \cdot X^-_{4^\ell f,g}(0) \\ &\leq 4^{-\ell} \cdot 2(X^+_{4^\ell f,g} - X^-_{4^\ell f,g}) \\ &\leq 2(X^+_{fg} - X^-_{fg}),\end{aligned}$$

where to get the first inequality we applied the assumption to the pair of flows $(4^\ell f, g)$.

To show (3.6) for the case $\ell = 0$, we will exhibit a *payment scheme* where each edge $e \in E$ will make a real valued payment $p(e)$. If $p(e)$ is negative for an edge e then we say that it receives payment. We will ensure our payment scheme satisfies the following.

PROPOSITION 3.1.

(a) The total payment over all edges is zero: $\sum_{e\in E} p(e) = 0$.

(b) Each edge $e \in E^+_{fg}$, *pays at most* $\mathrm{r}_e \cdot \vec{f}[e] \cdot \vec{g}[e]$; *so that the total payment from the edges of* E^+_{fg} *is at most* X^+.

(c) Each edge in $e \in E^-_{fg}$ *receives payment (i.e.* $p(e) < 0$*); the magnitude of this payment is at least*

$$\left(1 + \frac{\min\{|\vec{f}[e]|, |\vec{g}[e]|\}}{\max\{|\vec{f}[e]|, |\vec{g}[e]\}}\right) \cdot \mathrm{r}_e\, |\vec{f}[e] \cdot \vec{g}[e]|.$$

Let us complete the proof assuming that we are able to set up a payment scheme as claimed above. For edges $e \in E^-_{fg}(0)$, we have

$$\frac{\min\{|\vec{f}[e]|, |\vec{g}[e]|\}}{\max\{|\vec{f}[e]|, |\vec{g}[e]\}} \geq \frac{1}{2}.$$

Since the total payment is zero, we have

$$\sum_{e\in E^-_{fg}} \left(1 + \frac{1}{2}\right) \cdot \mathrm{r}_e\, |\vec{f}[e] \cdot \vec{g}[e]| \leq X^+_{fg}.$$

It follows that

$$X^-_{fg}(0) = \sum_{e\in E^-_{fg}(0)} \mathrm{r}_e \cdot |\vec{f}[e] \cdot \vec{g}[e]| \leq 2(X^+_{fg} - X^-_{fg}),$$

as claimed. This completes the proof of (3.6) assuming we can establish Proposition 3.1.

We now describe the payment scheme and prove Proposition 3.1. Consider the flow $\vec{f} + \vec{g}$. This flow can be decomposed into two flows $\vec{f}^*$ and $\vec{g}^*$ such that[1]

- $\vec{f}^*$ has the same source, target and value as $\vec{f}$;
- $\vec{g}^*$ has the same source, target and value as $\vec{g}$;
- $\vec{f}^* + \vec{g}^* = \vec{f} + \vec{g}$.
- $\vec{f}^*$ and $\vec{g}^*$ have the same orientation along each edge e (i.e., $\vec{f}^*[e] \cdot \vec{g}^*[e] \geq 0$).

Let $\vec{c}_f = f^* - f$ and $\vec{c}_g = g^* - g$; thus, $\vec{c}_f$ and $\vec{c}_g$ are circulations, and $\vec{c}_f = -\vec{c}_g$. The payment scheme $p : E \to \mathbb{R}$ is as follows.

$$\boxed{\begin{aligned}p(e) &= \mathrm{r}_e \cdot \vec{c}_f[e] \cdot (\vec{f}[e] - \vec{g}[e]) \\ &= \mathrm{r}_e \cdot \vec{c}_g[e] \cdot (\vec{g}[e] - \vec{f}[e]).\end{aligned}}$$

Proof of Proposition 3.1 To show part (a), observe that the total payment is

$$\sum_{e\in E} \mathrm{r}_e \cdot \vec{c}_f[e] \cdot (\vec{f}[e] - \vec{g}[e]).$$

The claim will follow if we show that

$$\sum_{e\in E} \mathrm{r}_e \cdot \vec{c}_f[e] \cdot \vec{f}[e] = 0; \tag{3.8}$$

$$\sum_{e\in E} \mathrm{r}_e \cdot \vec{c}_f[e] \cdot \vec{g}[e] = 0. \tag{3.9}$$

We will only show (3.8); to derive (3.9), replace f by g and use the fact that $\vec{c}_f = -\vec{c}_g$.

[1]This can be done by standard flow decomposition into paths. Let the source of $\vec{f}$ be s_f and the value of $\vec{f}$ be D_f; similarly, let s_g be the source of $\vec{g}$ and let D_g be the value of $\vec{g}$. Add a super-source s, an edge (s, s_f) carrying a flow of D_f, and an edge (s, s_g) carrying a flow of D_g. This creates a single flow from s to t of value $D_f + D_g$. Decomposing this flow into paths and combining paths that use edges (s, s_f) and (s, s_g) into $\vec{f}^*$ and $\vec{g}^*$, respectively, gives the required flows. Note that $\vec{f}^*$ and $\vec{g}^*$ are not necessarily unique.

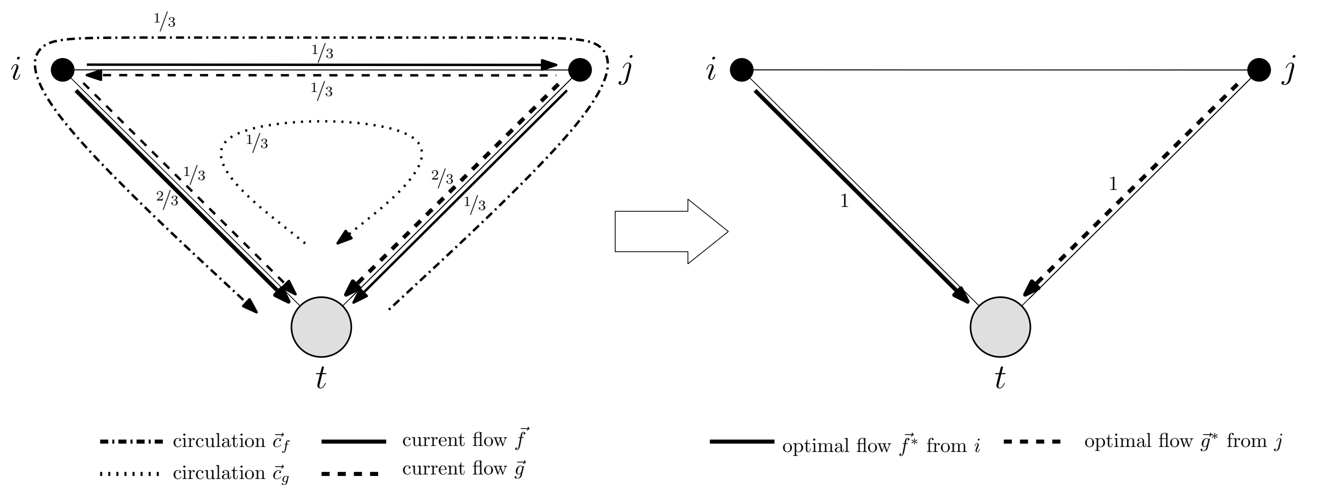

Figure 1: The circulation flows, current flows, and optimal flows for a simple network. The demands are $D_i = D_j = 1$ and all edges have resistance 1. Note that $\vec{f}^* = \vec{f} + \vec{c}_f$ and $\vec{g}^* = \vec{g} + \vec{c}_g$.

Recall that f satisfies Kirchhoff's laws. Thus there exist potentials $\langle \varphi[v] : v \in V \rangle$, such that for each edge e of the form (v, w), we have $\mathrm{r}_e \cdot \vec{f}[e] = \varphi[v] - \varphi[w]$. Substituting this in the LHS of (3.8), and collecting the contributions for each vertex, we obtain

$$\sum_{v \in V} \varphi[v] \left(\sum_{e \in \mathrm{Out}(v)} \vec{c}_f[e] - \sum_{e \in \mathrm{In}(v)} \vec{c}_f[e] \right).$$

Since $\vec{c}_f$ is a circulation, this quantity is zero (see (2.2)).

To show part (b), let $e \in E^+_{fg}$. We may assume that $|\vec{f}[e]| \geq |\vec{g}[e]|$, for otherwise we can argue by interchanging $\vec{f}$ and $\vec{g}$. Note that $\vec{f}[e]$, $\vec{g}[e]$, $(\vec{f} + \vec{g})[e]$, $\vec{f}^*$ and $\vec{g}^*$ have the same direction. We have two cases based on whether or not $\vec{c}_f[e]$ also has the same direction. If it has the same direction, then since $\vec{g}^*[e] = \vec{g}[e] + \vec{c}_g[e] = \vec{g}[e] - \vec{c}_f[e]$ and $\vec{g}^*[e]$ has the same direction as $\vec{g}[e]$, we have $|\vec{c}_f[e]| \leq |\vec{g}[e]|$, and

$$p(e) \leq \mathrm{r}_e \cdot \vec{g}[e] \cdot (\vec{f}[e] - \vec{g}[e]) \leq \mathrm{r}_e \cdot |\vec{f}[e] \cdot \vec{g}[e]|.$$

If $\vec{c}_f[e]$ is not in the same direction as the other currents on e, then $p(e)$ is negative (e receives payment!), and our claim holds because $\mathrm{r}_e \cdot \vec{f}[e] \cdot \vec{g}[e] \geq 0$.

To show part (c), fix an edge $e \in E^-_{fg}$. Again we may assume $|\vec{f}[e]| \geq |\vec{g}[e]|$. This time the flow f has opposite direction to g, but the same as $(\vec{f} + \vec{g})$ along edge e. In particular, the direction of $\vec{g}^*[e] = \vec{g}[e] + \vec{c}_g[e]$ is the same as $\vec{f}$'s, and hence opposite to $\vec{g}$'s. Thus, $\vec{c}_g[e]$ has the same direction as f, and $|\vec{c}_g[e]| \geq |\vec{g}[e]|$. Since $\vec{c}_f[e] = -\vec{c}_g[e]$, $\vec{c}_f$ and $\vec{f}$ have opposite directions along e, and the payment $p(e)$ is negative, with absolute value at least $\mathrm{r}_e \cdot |\vec{g}[e]| \cdot (|\vec{f}[e]| + |\vec{g}[e]|)$, as required. This completes the proof of Proposition 3.1

4 Naïve is no better

In this section, we show that the analysis in the previous section, showing that the solution of the naïve oblivious algorithm is always within a factor $O(\log n)$ of the optimum, cannot be improved.

THEOREM 4.1. *For every n, there exists an input instance (G, R, D), with $|V(G)| = n$, such that* $\textsc{Naïve}(G, R, D) = \Omega(\log n) \cdot \textsc{Opt}(G, R, D)$.

Proof. We will show the claim only for n of the form $2^d + 1$. A routine padding argument (e.g. new vertices with zero demands), will then establish the claim for all n.

Our graph will consist of a hypercube connected to a sink. More precisely, let H be the hypercube of dimension d, that is,

$$\begin{aligned} V(H) &= \{S : S \subseteq [d]\}; \\ E(H) &= \{\{S, T\} : |S \setminus T \cup T \setminus S| = 1\}. \end{aligned}$$

The undirected graph G is then defined by

$$\begin{aligned} V(G) &= V(H) \cup \{t\}; \\ E(G) &= E(H) \cup \{\{S, t\} : S \subseteq [d]\}. \end{aligned}$$

All resistances in G have value 1, and all vertices in $V(H)$ have unit demands. Theorem 4.1 will follow if we show the following two inequalities.

$$\textsc{Naïve}(G, R, D) \geq \frac{2}{9} d \cdot 2^d; \tag{4.10}$$

$$\textsc{Opt}(G, R, D) \leq 2^d. \tag{4.11}$$

To see (4.11), consider the solution that routes the demand at vertex S through the edge (S, t). The cost of this solution is 2^d.

Proof of (4.10). We will show that the Naïve oblivious algorithm imposes a load of at least $\frac{2}{3}$ on each hypercube edge. Then (4.10) follows immediately as there are $d2^{d-1}$ edges. By symmetry, all edges of the hypercube have the same load. So, it is enough to study any one edge, say, $\{\emptyset, \{1\}\}$, and show that it has a load of at least $\frac{2}{3}$.

Let H_0 be the subcube induced by the vertices in $V(H_0) = \{S : S \subseteq [d] \setminus \{1\}\}$ and H_1 be the subcube induced by the complement of $V(H_0)$. Consider the demand vector D_0 which assigns unit demands to vertices of H_0 and zero demand to all other vertices.

What does the optimal routing for demand D_0 look like? We view G as a resistive network and use Fact 2.2. By symmetry, all vertices of H_0 have the same potential, and also all nodes in H_1 have the same potential ϕ_1. Hence no current flows through the edges within the subcubes H_0 and H_1. Thus, the triangle induced by $\{\emptyset, \{1\}, t\}$, for example, can be analyzed in isolation using Ohm's law. It is easy to verify that the current through the edge $e = (\emptyset, \{1\})$ is $\frac{1}{3}$. By Fact 2.3, this current is precisely $\sum_{S \in V(H_0)} \vec{f_S}[e]$, where $\vec{f_S}$ denotes the optimal flow from S to t of unit value. Thus, $\sum_{S \in V(H_0)} |\vec{f_S}[e]| \geq \frac{1}{3}$. By symmetry,

$$\sum_{S \in V(H_1)} |\vec{f_S}[e]| \geq \frac{1}{3}, \text{ and, hence, } \sum_{S \in V(H)} |\vec{f_S}[e]| \geq \frac{2}{3}.$$

5 Oblivious isn't much better either

In this section, we prove that no oblivious routing algorithm can outperform the naïve oblivious algorithm more than quadratically in terms of worst-case hardness.

Theorem 5.1. *For every n, there is a graph G with resistances R, such that for any oblivious algorithm, there exists a demands set D such that the cost of the algorithm is $\Omega(\sqrt{\log n}) \cdot \mathrm{OPT}(G, R, D)$.*

Proof. Let G be the graph defined in the proof of Theorem 4.1, consisting of the hypercube of dimension d, plus a target vertex, t, incident to every other vertex. All edge resistances are 1.

Fix an oblivious routing algorithm. We will exhibit a distribution over demands D, such that, with positive probability, the algorithm's cost is $\Omega(\sqrt{\log n})\mathrm{OPT}(G, R, D)$.

To this end, let $L + 1$ denote an integer median for the path lengths of the given algorithm. To be more precise, let v be a uniformly randomly chosen vertex, and choose a random path from v to t with probability proportional to the flow sent by the oblivious routing algorithm along that path. Let L be the least positive integer such that with probability $\geq 1/2$, this path has length $\leq L + 1$. Observe that, by the minimality of L, the probability that the path has length $\geq L + 1$ is also at least $1/2$. (Our definition has lengths of $L + 1$ rather than L because the final edge of each path leaves the hypercube to reach the sink t.)

Let $d' = d - \lfloor d/2L \rfloor$. Choose a uniformly random subcube H of dimension d', and let D assign unit demands to every vertex of H.

Claim. *The expected competitive ratio is $\Omega(d/L + L)$.*

This expression is minimized when $L = d^{1/2}$, which completes the proof of the Theorem.

To see the Claim, first note that the optimal cost for these demands is at most $\frac{2(d-d')+1}{(d-d'+1)^2}|H|$. This cost bound can be achieved by splitting the demand from each vertex $v \in H$ equally along the $(d - d')$ paths of length 2 through v's neighbors in $V \setminus H$, as well as the direct edge from v to t.

Now consider the cost of the oblivious routing. The key insight is that, by the choice of d' and H, in expectation, a constant fraction of the demand is not routed outside of H except in the last step when it goes to the target. More precisely, for each path of length $\ell + 1 \leq L + 1$, the probability that this path includes any edge in one of the $(d - d')$ directions leaving H is (by a union bound) at most

$$(d - d')\frac{\ell}{d} \leq 1/2.$$

Hence the total expected flow along the $|H|$ edges joining H to t is at least $|H|/4$, which therefore contributes at least $|H|/16$ to the expected cost.

Now let us consider the cost of the long paths, of length $\ell \geq L + 1$. By the argument just given, with probability at least $1/2$, such a path does not leave H within its first L steps. Hence such paths contribute an expected load of at least $|H|L/4$ to the $|H|d'$ edges within the subcube, which contributes at least $|H|L^2/16d'$ to the expected cost.

Thus the competitive ratio is at least

$$\begin{aligned}
&\frac{1}{16}\frac{(d - d' + 1)^2}{1 + 2(d - d')\left(1 + \frac{L^2}{d'}\right)} \\
&\quad \geq \frac{1}{32}(d - d' + 1)\left(1 + \frac{L^2}{d}\right) \\
&\quad \geq \frac{1}{32}\frac{d}{2L}(1 + L^2/d) \quad \text{by definition of } d' \\
&\quad = \frac{1}{64}\left(\frac{d}{L} + L\right)
\end{aligned}$$

completing the proof. □

6 Conclusions

In this paper, we considered the problem of obliviously routing on a network with quadratic latency cost functions under the restriction that all routing requests are directed to the same target node. We showed that the naïve oblivious routing algorithm achieves a competitive ratio of $O(\log n)$ on a network of n nodes. This result can be strengthened to $O(\log k)$, whre k denotes the number of different sources of the network. We then analyzed the special case of routing on a hypercube and showed that the naïve algorithm can perform no better (up to constant factors). Furthermore, using the same hypercube example, we showed that no oblivious algorithm can achieve a competitive ratio better than $\Omega(\sqrt{\log n})$.

A challenging question for future research is whether the naïve algorithm that we analyzed also obtains a polylogarithmic competitive ratio in the multi-commodity case. We believe that this is indeed the case but resolving this question seems to require new techniques as it is not possible to exploit the relationship between quadratic cost networks and resistive networks.

Improving the upper bound for the single-commodity case runs into the same obstacles. Therefore it might be a more promising approach to try to improve the lower bound. In particular, it would be interesting to improve the lower bound in the multi-commodity case. Currently, the best lower bound for the single-commodity and multi-commodity case are identical and it seems not clear how to use more commodities to strengthen the lower bound.

Another interesting problem is to determine cost-measures for which the naïve algorithm performs well. It (trivially) works for the $\|\cdot\|_1$-norm of the link loads and it is conjectured to give a competitive ratio of $O(\log n)$ for the $\|\cdot\|_\infty$-norm (congestion) in the single-target case. Does an $O(\log n)$ bound hold for any ℓ_p-norm of link loads?

References

[1] D. Applegate and E. Cohen, *Making intra-domain routing robust to changing and uncertain traffic demands: Understanding fundamental tradeoffs*, in Proceedings of the ACM Symposium on Communications Architectures & Protocols (SIGCOMM), 2003, pp. 313–324.

[2] B. Awerbuch, Y. Azar, E. F. Grove, M.-Y. Kao, P. Krishnan, and J. S. Vitter, *Load balancing in the Lp norm.*, in Proceedings of the 36th IEEE Symposium on Foundations of Computer Science (FOCS), 1995, pp. 383–391.

[3] Y. Azar, E. Cohen, A. Fiat, H. Kaplan, and H. Räcke, *Optimal oblivious routing in polynomial time*, in Proceedings of the 35th ACM Symposium on Theory of Computing (STOC), 2003, pp. 383–388.

[4] P. G. Doyle and J. L. Snell, *Random Walks and Electric Networks*, Mathematical Association of America, 1984.

[5] A. Gupta, M. T. Hajiaghayi, and H. Räcke, *Oblivious network design*, in Proceedings of the 17th ACM-SIAM Symposium on Discrete Algorithms (SODA), 2006, pp. 970–979.

[6] M. T. Hajiaghayi, J. H. Kim, F. T. Leighton, and H. Räcke, *Oblivious routing in directed graphs with random demands*, in Proceedings of the 37th ACM Symposium on Theory of Computing (STOC), 2005, pp. 193–201.

[7] M. T. Hajiaghayi, R. D. Kleinberg, F. T. Leighton, and H. Räcke, *Oblivious routing on node-capacitated and directed graphs*, in Proceedings of the 16th ACM-SIAM Symposium on Discrete Algorithms (SODA), 2005, pp. 782–790.

[8] C. Harrelson, K. Hildrum, and S. B. Rao, *A polynomial-time tree decomposition to minimize congestion*, in Proceedings of the 15th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA), 2003, pp. 34–43.

[9] H. Räcke, *Minimizing congestion in general networks*, in Proceedings of the 43rd IEEE Symposium on Foundations of Computer Science (FOCS), 2002, pp. 43–52.

[10] T. Roughgarden and É. Tardos, *How bad is selfish routing?*, Journal of the ACM, 49 (2002), pp. 236–259.

[11] L. G. Valiant and G. J. Brebner, *Universal schemes for parallel communication*, in Proceedings of the 13th ACM Symposium on Theory of Computing (STOC), 1981, pp. 263–277.