CS 598CSC: Topics in Graph Algorithms	**Lecture date: 09/24/2024**
Instructor: Chandra Chekuri	**Scribe: CC**

1 Introduction to Oblivious Routing

Oblivious routing originated in parallel and distributed computing and also has connections to online algorithms. Consider a communication network represented by a graph $G = (V, E)$ with edge capacities $c : E \to \mathbb{R}_+$. This network is used to route/satisfy demands between source-sink pairs. In most networks the actual demands are not known in advance and come and go in an online fashion. Which route/path(s) should the demand for some specific pair (s, t) be routed on? It depends on other demands. To feasibly route a given set of demands in the network we need to essentially solve a multicommodity flow problem which helps to balance the network's capacity among many competing pairs. This is a non-trivial computational problem. There are also various issues on how to specify the end-to-end routes and how to implement the routing. Some of these practical issues have motivated interesting theoretical problems, some of which have led to fundamental breakthroughs that seem quite far from the original motivation. Oblivious routing is one such problem.

The goal in oblivious routing, in some sense, is to bridge the static setting where the demands are all fully known to the fully online setting where we specify a route to a new demand when it arrives. In oblivious routing, for each pair of vertices (u, v) we specify a *static* probability distribution over the paths from u to v in G (denoted by $\mathcal{P}_{u,v}$). This may seem wild because there could be an exponential number of paths in $\mathcal{P}_{u,v}$. For now, think of a distribution over a small number of paths or an implicit and compact specification. When a demand to route $D(u, v)$ units of flow between u and v arrives *online*, the scheme will pick a path $P \in \mathcal{P}_{u,v}$ according to the static distribution and route the flow along that path. Alternatively, we can also think of sending the $D(u, v)$ demand fractionally along the paths in proportion to the probability. This scheme explains the name *oblivious* because the distributions are required to be computed before knowing any of the actual demands. Clearly, if we knew the entire set of demands in advance we can compute an optimal routing via some sort of multicommodity flow computation. How well can an oblivious routing do when compared to a static routing? How do we measure the quality? And do good oblivious routings exist? The initial paper that motivated this is due to Valiant and Brebner [VB81] who showed that good deterministic oblivious routings exist in special classes of graphs such as hypercubes. Later, the need for randomization in the sense of picking paths randomly as we described was realized.

How do we measure the quality of oblivious routing? There are several ways but historically the main one, and the one which has had the most theoretical impact, is the one that focused on *congestion*. We first digress towards defining routable demands and some properties before formally defining the quality of oblivious routing.

1.1 Routable demands in a network

We will work with directed graphs since flow is more naturally defined in such graphs and later indicate the changes needed for undirected graphs. Let $G = (V, E)$ be a directed graph with edge capacities $c : E \to \mathbb{R}_+$. It is common in networking literature to use the notion of demand matrices. $D \in \mathbb{R}^{n \times n}$ is a matrix that specifies the demand between every ordered pair of vertices (u, v) (in networking they often use indices 1 to n to specify vertices and hence the use of matrix notation).

Definition 1. *A demand matrix D is routable in G if there is a multicommodity flow in G that routes $D(u, v)$ amount of demand for every pair of vertices (u, v). D is said to be routable with congestion ρ where $\rho > 0$ is a non-negative scalar if D is routable in G where edge capacities are multiplied by ρ.*

Thus $\rho = 1$ corresponds to a routable demand matrix. The following claim is easy but quite useful.

Claim 1. *If D and D' are routable matrices then so is $\lambda D + (1 - \lambda)D'$ for any scalar $\lambda \in [0, 1]$. Let $\mathcal{D}_G$ be the set of all demand matrices that are routable in G. Then $\mathcal{D}_G$ is a convex set in $\mathbb{R}^{n^2}$.*

Note that the set of all demand matrices routable in G with congestion ρ, for any fixed ρ, is also convex. Given a demand matrix D we can efficiently check if D is routable in G via solving an LP for multicommodity flow. Thus, we have a membership oracle for the convex set $\mathcal{D}_G$. The flow LP is nice because the LP is also linear in $D(u, v)$ values. Thus, we can claim something stronger. There is an efficient *optimization* oracle over $\mathcal{D}_G$. What does this mean? That is, given weights $w(u, v)$ for each pair (u, v) of vertices we can efficient solve the problem $\sum_{u,v} w(u, v)D(u, v)$ such that $D \in \mathcal{D}_G$.

Exercise 1. *Argue formally why we have an efficient optimization oracle over $\mathcal{D}_G$.*

By the equivalence of optimization and separation, we have an efficient separation oracle over $\mathcal{D}_G$! That is, there is an efficient algorithm, that given $D \in \mathbb{R}^{n^2}$, either outputs that $D \in \mathcal{D}_G$ or outputs a separating hyperplane that separates D (viewed as a point in n^2 dimensional space) from the convex set $\mathcal{D}_G$. This is a convoluted but easy way of deriving a useful fact. You may suspect that there would be more direct ways of deriving this. The following is often referred to as the Japanese theorem on multicommodity flow which can be derived from LP duality which we saw when discussing sparsest cut.

Exercise 2. *Let D be a demand matrix. Then D is routable in G iff* for all *length functions $\ell : E \to \mathbb{R}_+$,*

$$\sum_{e \in E} c_e \ell(e) \geq \sum_{(u,v) \in V \times V} D(u, v)\, dist_\ell(u, v)$$

where $dist_\ell(u, v)$ is the shortest path distance from u to v induced by edge lengths given by ℓ.

Thus, to prove that D is *not* routable in G it suffices to produce one length function $\ell : E \to \mathbb{R}_+$ that violates the inequality in the preceding exercise. Thus, one can view $\mathcal{D}_G$ as being defined by an infinite set of linear inequalities, one for each non-negatively length function.

Undirected graphs: For the most part we will be interested in undirected graphs. Typically we will consider demands as unordered pairs uv, that is, we specify a demand matrix D as a symmetric matrix with the understanding that we wish to route $D(uv)$ demand for the pair uv. Undirected multicommodity flow is somewhat clunky to define compactly (but it can be done by bi-directing the edges and capping the total flow on both copies of the edges by the original capacity $c(e)$) but easy to specify via path flow. For this reason we defined things via directed graphs. We will see that the optimal oblivious routing can be computed in directed graphs as well but the optimal value can be large unlike in undirected graphs.

1.2 Oblivious routing to minimize congestion

Recall that in oblivious routing we wanted to specify a probability distribution over paths for each pair (u, v). One can view such a probability distribution as a flow of one unit from u to v where the flow on a path $P \in \mathcal{P}_{u,v}$ is equal to the probability of P. Thus, one compact way to represent a probability distribution over paths is via a unit flow via flow values on edges which can be specified using only m numbers. Note that an edge-based flow does not uniquely specify a path-based flow so we are losing information by using the compact representation but it is helpful in some computational situations.

Definition 2. *An edge-based oblivious routing for a network $G = (V, E)$ is a collection of unit flows in G, one for each pair $(u, v) \in V \times V$, specified via edge-based flows. A path-based oblivious routing for a network $G = (V, E)$ is similar where the unit flow is specified via a path based flow.*

We will use $f_e^{(u,v)}$ to denote the amount of flow on edge e for pair (u, v) specified by the oblivious routing; note that this is a number in $[0, 1]$. Whether the oblivious routing is specified via edge-based flow or path-based flow, this quantity is well defined.

Definition 3. *Let D be a demand matrix. The congestion incurred for D via the oblivious routing specified by values, $f_e^{(u,v)}$, $e \in E, (u, v) \in V \times V$ is $\max_{e \in E} \frac{\sum_{u,v} D(u,v) f_e^{(u,v)}}{c_e}$. We say that D is routed with congestion ρ by the oblivious routing.*

Definition 4. *Let $G = (V, E)$ be a network with edge capacities $c : E \to \mathbb{R}_+$. The congestion of an oblivious routing is the smallest $\rho \geq 1$ such every routable demand $D \in \mathcal{D}_G$ is routed with congestion at most ρ by the oblivious routing. An optimal oblivious routing for G is the one which achieves the smallest ρ among all oblivious routings.*

In other words, we are asking how much should we scale up the capacities of G so that any demand matrix D that is routable in G (statically with full knowledge of D), can be routed in G via the oblivious routing.

Exercise 3. *Prove that every graph G admits an n^2-congestion oblivious routing.*

Some fundamental results on oblivious routing: In roughly chronological order.

- Harald Räcke [Rac02], in a breakthrough result, proved that every undirected graph admits an oblivious routing with congestion $O(\log^3 n)$. His initial proof used an optimal algorithm for sparsest cut and hence did not lead to an efficient algorithm to construct the oblivious routing. Soon after, [BKR03] and [HHR03] obtained efficient algorithms with the [HHR03] obtaining a congestion of $O(\log^2 n \log \log n)$. These results are based on a hierarchical expander decomposition of the graph which results in a compact cut representation of every graph. It is a non-trivial a fundamental structural result on graphs which has found many applications in fast graph algorithms recently (see [GRST21] for instance).

- Azar et al [ACF$^+$03] proved, via a simple idea in retrospect, that the optimal oblivious routing can be computed efficiently via LP techniques even for directed graphs. We will see it in the next section. Note that being able to compute an optimum oblivious routing for any given graph does not tell us an easy to understand universal bound such as $\log n$ or $\sqrt{n}$.

- Oblivious routing in directed graphs requires congestion $\Omega(\sqrt{n})$ [ACF$^+$03]; this lower bound holds even for the restricted case of single-source demands. A similar lower bound was shown to hold also for undirected graphs with node capacities [HKRL07]. More recently, the lower bound for directed graphs has been improved to $\Omega(n)$ [EMPS16].

- In another breakthrough, Räcke [Räc08], made a beautiful and surprising connection between oblivious routing and probabilistic tree embeddings for metric distortion, and obtained an optimal $O(\log n)$-congestion tree-based oblivious routing scheme. Via this algorithm he obtained an $O(\log n)$-approximation for the minimum bisection problem in graphs. In addition the tree-based oblivious routing has a number of applications in network design including some recent work of the instructor [CJ24].

2 Efficient Algorithm to Compute an Optimal Oblivious Routing

In this section we prove that one can find an optimal edge-flow based oblivious routing in polynomial time. This is due to [ACF$^+$03] and is quite simple in retrospect. We will work with directed graphs since it is easier to define edge-based flows. Recall that any oblivious routing can be specified by variables $f_e^{(u,v)}, e \in E, (u,v) \in V \times V$ such that for any pair (u,v) the variables $f_e^{(u,v)}, e \in E$ define a unit flow from u to v in G. To find an oblivious routing with minimum congestion we write the following LP essentially following the definitions. (Recall that $\mathcal{D}_G$ is the set of all demand matrices that are routable in G.)

$$\begin{aligned}
\min \rho & \\
\text{s.t} & \\
f_e^{(u,v)}, e \in E & \quad \text{is a unit flow from } u \text{ to } v \quad \forall (u,v) \in V \times V \\
\sum_{(u,v) \in V \times V} D(u,v) f_e^{(u,v)} & \leq \rho c_e \quad \forall D \in \mathcal{D}_G \\
f_e^{(u,v)} & \geq 0 \quad \forall e \in E, \forall (u,v) \in V \times V
\end{aligned}$$

Remark 2. *We did not write down the full description of the LP to keep the high-level ideas clean. It is easy to write down linear constraints that capture the fact that $f_e^{(u,v)}, e \in E$ define a unit flow from u to v for each (u,v). We leave this as an exercise.*

The only catch in solving the LP is that it has an *infinite* number of constraints but notice that the number of variables is large but polynomial; a total of mn^2 variables. We can use the Ellipsoid method if we have an efficient separation oracle. We understand what is required of the separation oracle. Suppose we are given a set of numbers $y_e^{(u,v)}, e \in E, (u,v) \in V \times V$ and a number ρ'. The oracle needs to correctly verify that these satisfy the constraints or output a constraint that is violated. It is easy to check that the values define a unit flow for each pair. In fact these constraints are polynomial in number. Thus, the interesting constraints are about the demand matrices. Suppose we fix an edge e. To know if there is any demand matrix $D \in \mathcal{D}_G$ such that $\sum_{(u,v)\in V\times V} D(u,v)y_e^{(u,v)} \geq \rho' c_e$ we can solve an LP over $\mathcal{D}_G$! The objective weights are $y_e^{(u,v)}, (u,v) \in V \times V$. We saw that we can optimize/maximize linear objectives over $\mathcal{D}_G$ efficiently. And thus, we can check each edge to see if there is any demand matrix which violates the congestion constraint for the choice of ρ'.

Exercise 4. *Verify that there is an efficient separation oracle for the LP to compute an optimum oblivious routing in a given network G.*

Of course the resulting algorithm is not every efficient but it shows the power of the Ellipsoid method and working with large implicit LPs fearlessly. One can use multiplicative weight updates and other techniques to obtain fast approximation algorithms for some of these problems and we will see this later in the course.

The preceding technique to design optimum oblivious routing is fairly general. We do not need to consider the full set of demand matrices $\mathcal{D}_G$. As long as we have a nice convex set of demand matrices that we can optimize over, we can find an optimum oblivious routing when restricted to those demand matrices. See [AC03] for an empirical evaluation of this idea. See [Che07] for some pointers to work on dealing with uncertainty in the demand matrix (these articles are somewhat dated but it is possible to trace more recent literature via citations).

3 Tree-based Oblivious Routing via Duality and Low Stretch Trees

In this section we will prove that in undirected graphs there is always an oblivious routing with congestion $O(\log n)$ which is an optimal bound. Räcke prove this result via an elegant connection to tree embeddings for distance preservation. The bound, the connection, and the tree-based aspect are all important. As we discussed previously, probabilistic approximation of a graph by spanning trees for distances incurs an $O(\log n \log\log n)$ distortion while we can obtain an optimal $O(\log n)$ distortion if we allow dominating tree metrics. It is easier to understand the ideas in Räcke's proof, and in particular the notation, by working with spanning trees than with general hierarchical tree representations of graphs (in some cases the spanning tree result is useful/needed). The difference in the distortion is not significant for our purposes here and we will use these results in a black box fashion. At the end of the section we sketch how to use dominating tree metrics rather than spanning trees. We let $\alpha(n)$ denote the best bound that we can obtain for approximating the distances in a n-node graph $G = (V, E)$ with edge lengths $\ell : E \to \mathbb{R}_+$ by a probability distribution over spanning trees. Currently we have $\alpha(n) = O(\log n \log\log n)$ [ABN08] and it is conjectured that $\alpha(n) = O(\log n)$. A simple implication of this result is the following claim.

Claim 3. *Let $G = (V, E)$ be a graph with non-negative edge lengths $\ell : E \to \mathbb{R}_+$. Let $w : E \to \mathbb{R}_+$ be any set of non-negative edge weights. Then there is a spanning tree $T \in \mathcal{T}_G$ such that $\frac{\sum_{uv\in E} w(uv)\, dist_T(u,v)}{\sum_{e\in E} w(e)\ell(e)} \leq \alpha(n)$ where $dist_T(u,v)$ is the length of the unique path from u to v in T.*

Proof. Recall that there is a probability distribution $p : \mathcal{T}_G \to [0,1]$ such that for every pair of vertices $u, v \in V$, $E[\text{dist}_T(u,v)] \leq \alpha(n)\text{dist}_G(u,v)$ where distances are induced by the edge lengths $\ell : E \to \mathbb{R}_+$. Now fix any weight function w over pairs of vertices. Then by linearity of expectation, $E[\sum_{u,v} w(uv)\text{dist}_T(u,v)] \leq \alpha(n)\sum_{u,v} w(uv)\text{dist}_G(u,v)$. Thus, there is at least one tree T such that $\sum_{u,v} w(uv)\text{dist}_T(u,v) \leq \alpha(n) \sum_{u,v} w(uv)\text{dist}_G(u,v)$. Now consider the case when the support of the weights w is only on the edges of G, that is $w(uv) = 0$ if $uv \notin E$. In this case $\text{dist}_G(u,v) \leq \ell(u,v)$. And $\text{dist}_T(u,v)$ is the shortest path length along the path in T. Thus, we obtain $\sum_{uv \in E} w(uv)\text{dist}_T(u,v) \leq \alpha(n) \sum_{uv \in E} w(uv) d_G(u,v) \leq \alpha(n) \sum_{uv \in E} w(uv)\ell(u,v)$. ■

Tree based oblivious routing: We will consider a specific type of oblivious routing. Consider *any* probability distribution p over the spanning trees of G. We observe that this distribution induces an oblivious routing. How? Fix any pair u, v. For each tree $T \in \mathcal{T}_G$ there is unique path $P_T(u,v)$ between u and v in G. Thus, the probability distribution over $\mathcal{T}_G$ induces a probability distribution over paths between u and v. Note that the distribution over trees induces a distribution for every pair of vertices simultaneously. We call such an oblivious routing scheme a tree-distribution-based oblivious routing (this is not necessarily a standard terminology). It is a restricted class of oblivious routings. This class of oblivious routing is particularly nice, at least from a theoretical point of view. We sample a tree from the distribution and whenever a demand arrives we simply route it along the unique path in the tree. Two natural questions arise.

- How good is the best tree-distribution-based oblivious routing?
- Can the best tree-distribution-based oblivious routing be efficiently computed?

Räcke showed that tree-distribution-based oblivious routings have a nice structural property that allow one to characterize the congestion in a simple way. To understand this, we set up some notation. Let T be a spanning tree and consider an edge $e \in E_T$. $T - e$ induces a partition of the vertex set into two sets $(S, V - S)$. We define the *load* on e due to T, denoted by $L(T,e)$ as $c(\delta(S))$. Why? Think of all the edges crossing the cut $(S, V - S)$. For each of those edges $e' = (s,t) \in \delta(S)$, the path from s to t in T has to go via e, and hence if we want to route demands corresponding to edges then the congestion on e will be $L(T,e)/c(e)$. We will let $L(T,e) = 0$ if $e \notin T$.

Definition 5. *Let $p : \mathcal{T}_G \to [0,1]$ be a probability distribution that induces an oblivious routing. For edge e let $L(e) = \sum_T p(T)L(T,e)$ be the expected load on e and let $\rho(e) = L(e)/c(e)$ be the expected congestion on e.*

The simple yet important observation is the following which characterizes the quality of the oblivious routing based on the expected congestion of the edges. Note that there is no reference to routable demands in this characterization and the proof will make it clear as to why that is the case.

Claim 4. *Let $p : \mathcal{T}_G \to [0,1]$ be a probability distribution that induces an oblivious routing. The congestion of this oblivious routing is at most* $\max_{e \in E} \rho(e)$.

Proof. Let $D \in \mathcal{D}_G$ be any routable demand in G. Fix some routing of D in G and let $x(e)$ be the total flow on edge e in the routing. Clearly $x(u,v) \leq c(u,v)$ for all $(u,v) \in E$. Let $L'(T,e)$ the total flow on e induced by the oblivious routing scheme for routing D on tree T. It suffices to prove that $L'(T,e) \leq L(T,e)$ for each e which would imply that $L'(e) = \sum_T p_T L'(T,e) \leq L(e)$. This is easy for the following reason. $L'(T,e)$ is equal to the sum of demands for vertex pairs that are separated in $T - e$. In other words if $T - e$ induces the cut $(S, V - S)$ then $L'(T,e) = \sum_{(u,v):|S \cap \{u,v\}|=1} D(u,v)$. Since D is routable, the total demand across the cut $(S, V - S)$ is at most $c(\delta(S))$ but the load $L(T,e)$ is defined as $c(\delta(S))$. ■

LP for optimum tree-based oblivious routing: The preceding characterization of the congestion allows us to write an LP to find the best tree-distribution based oblivious routing scheme. We will use variable x_T to denote the probability that $T \in \mathcal{T}_G$ is chose in the probability distribution. We use ρ to denote the congestion that we wish to minimize. We write down constraints that express x as a probability distribution and to express the load on each edge being bound by $\rho c(e)$.

$$\begin{aligned}
\min \rho & \\
\text{s.t} & \\
\sum_{T \in \mathcal{T}_G} x_T &= 1 \\
\sum_{T} x_T L(T, e) &\leq \rho c(e) \quad \forall e \in E \\
x_T &\geq 0 \quad \forall T \in \mathcal{T}_G
\end{aligned}$$

We write its dual below.

$$\begin{aligned}
\max \beta & \\
\text{s.t} & \\
\sum_{e \in E} c(e) z_e &= 1 \\
\sum_{e \in T} L(T, e) z_e &\geq \beta \quad \forall T \in \mathcal{T}_G \\
z_e &\geq 0 \quad \forall e \in E
\end{aligned}$$

It is useful to rewrite the dual in an equivalent ratio form.

$$\max_{z \in \mathbb{R}_+^m} \min_{T \in \mathcal{T}_G} \frac{\sum_{e \in T} L(T, e) z_e}{\sum_{e \in E} c(e) z_e}.$$

Exercise 5. *Argue that the max-min formulation is equivalent to the LP dual.*

Bounding the dual value: The main "observation" is that Claim 3 implies that the optimal dual value is $\alpha(n)$ which corresponds to the bound for tree-based distance approximation! Suppose this was true then we have shown that there exists a tree-distribution based oblivious routing with congestion $O(\log n \log \log n)$. We now proceed to prove this observation.

Lemma 5. *The optimal dual value is at most $\alpha(n)$.*

Proof. Fix any optimum solution z^* to the dual program

$$\max_{z \in \mathbb{R}_+^m} \min_{T \in \mathcal{T}_G} \frac{\sum_{e \in T} L(T, e) z_e}{\sum_{e \in E} c(e) z_e}.$$

We observe, by using the definition of $L(T, e)$ and interchaning the order of summation, that

$$\sum_{e \in T} L(T, e) z_e = \sum_{uv \in E} c(uv) \sum_{e \in P_T(u,v)} z_e$$

where $P_T(u,v)$ is the path from u to v in T. Note that $\sum_{e \in P_T(u,v} z_e$ can be thought of as the length of the path from u to v in T according to lengths given by z.

We wish to use Claim 3. We think of $c(e)$ as a weight $w(e)$ and think of z_e as length $\ell(e)$. With this interpretation and the preceding paragraph, we see that the dual can be written as

$$\max_{\ell \in \mathbb{R}_+^m} \min_{T \in \mathcal{T}_G} \frac{\sum_{uv \in E} w(uv) \mathrm{dist}_T(u,v)}{\sum_{uv \in E} w(u,v) \ell(u,v)}.$$

By Claim 3, this is at most $\alpha(n)$. ■

3.1 Going beyond spanning trees

To obtain the optimal $O(\log n)$-bound for oblivious routing we can use the optimum result for embedding a finite metric space on n points into a distribution over dominating tree metrics [FRT03]. This requires some notational work to set up but the proof is essentially the same as it is for spanning trees. Even though it is possible to work with trees that include Steiner vertices, the notation is still cumbersome. For this reason we go one step further than spanning trees and work with trees that only contain vertices in V and these suffice for most applications.

Path-mapped spanning trees: Given G let $\mathcal{T}'_G$ be the set of all spanning trees over the complete graph on V — note that a tree T may now have an edge uv where uv is not an edge of G. When working with metric distortion we only keep track of the length of such edges but for the oblivious routing application we need more information. Thus, for each $T = (V, E_T) \in \mathcal{T}_G$ we also maintain a map $M_T : E_T \to \mathcal{P}_G$ where $\mathcal{P}_G$ is the set of all paths/walks in G. For the map to make sense, we require that M_T has the following property: it maps an edge $uv \in E_T$ to a path $p \in \mathcal{P}_G(u,v)$ where $\mathcal{P}_G(u,v)$ is the set of all u-v paths in G. Thus, given any $s,t \in V$ we can map the unique s-t path in T to an s-t walk in G by using the map M_T. We use $M_T(s,t)$ to denote this walk. To make it easier to understand, we call the pair (T, M_T) a *path-mapped spanning tree* (pmst for short). We overload notation and use $\mathcal{T}'_G$ to denote the set of all pmsts of G. Note that this is not standard terminology in the literature but we are using it to make it easier to describe and understand. An important thing to note is the following: given (T, M_T) an edge $e' \in E$ (of the original graph G) can be in several paths associated with edges of T.

We now consider probability distributions over pmsts for a given graph G. We first consider such mappings for approximating distances.

Path-mapped spanning trees for approximating distances: Given a graph $G = (V, E)$ with edge lengths $\ell : E \to \mathbb{R}_+$ a pmst (T, M_T) naturally defines a dominating tree metric for the metric (V,d) induced by the shortest path distances. For each edge $e \in T$ we associate a length equal to the length of the path $M_T(e)$. Thus for an pair s,t, we have $\mathrm{dist}_T(s,t) = \ell(M_T(s,t))$ where $M_T(s,t)$ is a path in G. Clearly $\ell(M_T(s,t)) \geq \mathrm{dist}_G(s,t)$. It is not too difficult to prove that the $O(\log n)$ approximation of a n-point metric (V,d) by dominating tree metrics [FRT03] that we saw earlier, implies the following.

Theorem 6. *For any graph $G = (V,E)$ with non-negative edge lengths $\ell : E \to \mathbb{R}_+$ there is a probabilisty distribution $\mathcal{D}$ over $\mathcal{T}'_G$ such that for any pair of vertices u,v, $E[\ell(M_T(u,v)] \leq O(\log n)\, dist_\ell(u,v)$.*

We leave the proof as an exercise to the reader. One can then obtain a natural generalization of Claim 3.

Claim 7. *Let $G = (V, E)$ be a graph with non-negative edge lengths $\ell : E \to \mathbb{R}_+$. Let $w : E \to \mathbb{R}_+$ be non-negative weights on edges. Then there is a pmst $(T, M_T) \in \mathcal{T}'_G$ such that $\frac{\sum_{uv \in E} w(uv)\ell(M_T(u,v))}{\sum_{uv \in E} w(uv)\ell(u,v))} \leq O(\log n)$.*

Path-mapped spanning trees for oblivious routing: Let $G = (V, E)$ be a graph with edge capacities $c : E \to \mathbb{R}_+$. Fix a pair of vertices (s, t). Each pmst (T, M_T) defines a path $M_T(s, t)$ for pair (s, t). And hence any probability distribution over $\mathcal{T}'_G$ naturally defines an oblivious routing for G.

As we did with spanning trees, we can characterize the congestion induced by a probability distribution over $\mathcal{T}'_G$. For a given edge $e \in E$ and a tree T we would like to characterize load $L(T, e)$ induced by using T. Note that e can be in multiple paths associated with the edges of T. For each edge $e' \in E_T$ we let $(S_{e'}, V - S_{e'})$ be the partition of V induced by $T - e'$. We define

$$L(T, e) = \sum_{e' \in E_T : M_T(e') \ni e} c(\delta_G(S_{e'}))$$

and observe that this corresponds to the load induced on e by the pmst-based oblivious routing scheme. Given a probability distribution $p : \mathcal{T}'_G \to [0, 1]$ the expected load on $e \in E$ can be naturally defined as: $L(e) = \sum_T p_T L(T, e)$. This defines the congestion on $e \in E$, $\rho(e)$, as $L(e)/c(e)$ and the max congestion of the scheme as: $\rho = \max_{e \in E} \rho(e)$.

We obtain the following claim, similar to Claim 4.

Claim 8. *Let $p : \mathcal{T}'_G \to [0, 1]$ be a probability distribution that induces an oblivious routing. The congestion of this oblivious routing is at most $\max_{e \in E} \rho(e)$.*

The rest of the proof is very similar to the one for spanning trees where we write an LP relaxation to find the best probability distribution to minimize the congestion of the resulting oblivious routing (based on the preceding claim's characterization). We bound the dual's objective using Claim 7. This yields the optimal $O(\log n)$-congestion tree-based oblivious routing. We leave the formal derivation as a useful exercise for the reader.

3.2 Computational aspects

The proof of tree-based oblivious routings was based on duality. It is not immediately clear that it leads to an efficient algorithm but it does. As one would expect, we need an efficient algorithm for the dual separation oracle. This is the problem of approximating distances in the graph by spanning trees/pmsts and we have seen efficient algorithms for it. It is not obvious that we can use such approximate algorithms in the dual but there are standard techniques via the the multiplicative weight update (MWU) method and related ideas. See [Räc08].

References

[ABN08] Ittai Abraham, Yair Bartal, and Ofer Neiman. Nearly tight low stretch spanning trees. In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 781–790. IEEE, 2008.

[AC03] David Applegate and Edith Cohen. Making intra-domain routing robust to changing and uncertain traffic demands: understanding fundamental tradeoffs. In *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '03, page 313–324, New York, NY, USA, 2003. Association for Computing Machinery.

[ACF$^+$03] Yossi Azar, Edith Cohen, Amos Fiat, Haim Kaplan, and Harald Racke. Optimal oblivious routing in polynomial time. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 383–388, 2003.

[BKR03] Marcin Bienkowski, Miroslaw Korzeniowski, and Harald Räcke. A practical algorithm for constructing oblivious routing schemes. In *Proceedings of the fifteenth annual ACM symposium on Parallel algorithms and architectures*, pages 24–33, 2003.

[Che07] Chandra Chekuri. Routing and network design with robustness to changing or uncertain traffic demands. *ACM SIGACT News*, 38(3):106–129, 2007.

[CJ24] Chandra Chekuri and Rhea Jain. Approximation algorithms for network design in non-uniform fault models, 2024.

[EMPS16] Alina Ene, Gary Miller, Jakub Pachocki, and Aaron Sidford. Routing under balance. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 598–611, 2016.

[FRT03] Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 448–455, 2003.

[GRST21] Gramoz Goranci, Harald Räcke, Thatchaphol Saranurak, and Zihan Tan. The expander hierarchy and its applications to dynamic graph algorithms. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2212–2228. SIAM, 2021.

[HHR03] Chris Harrelson, Kirsten Hildrum, and Satish Rao. A polynomial-time tree decomposition to minimize congestion. In *Proceedings of the fifteenth annual ACM symposium on Parallel algorithms and architectures*, pages 34–43, 2003.

[HKRL07] Mohammad Taghi Hajiaghayi, Robert D. Kleinberg, Harald Räcke, and Tom Leighton. Oblivious routing on node-capacitated and directed graphs. *ACM Trans. Algorithms*, 3(4):51–es, nov 2007.

[Rac02] Harald Racke. Minimizing congestion in general networks. In *The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings.*, pages 43–52. IEEE, 2002.

[Räc08] Harald Räcke. Optimal hierarchical decompositions for congestion minimization in networks. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 255–264, 2008.

[VB81] Leslie G Valiant and Gordon J Brebner. Universal schemes for parallel communication. In *Proceedings of the thirteenth annual ACM symposium on Theory of computing*, pages 263–277, 1981.