# LP Duality, Lagrangian Relaxations, and Oblivious Routing

Kent Quanrud

September 30, 2020

## 1 LP Duality

We have previously discussed **linear programs (LPs)** particular for designing approximation algorithms for NP-Hard problems. Here we will consider to simple and canonical classes of LP, and how they relate via *LP duality*.

**Packing LPs** A **packing LP** is a linear program of the form

$$\max\langle b, x\rangle \text{ over } x \in \mathbb{R}^n_{\geq 0} \text{ s.t.} Ax \leq c. \tag{P}$$

where $A \in \mathbb{R}^{m\times n}_{\geq 0}$, $b \in \mathbb{R}^n_{>0}$, and $c \in \mathbb{R}^m_{>0}$ all have nonnegative coefficients. Abusing notation, we also let (P) denote the optimal value of the LP (P).

**Covering LPs** A **covering LP** is a linear program of the form

$$\min\langle c, y\rangle \text{ over } y \in \mathbb{R}^m_{\geq 0} \text{ s.t. } A^T y \geq b, \tag{C}$$

where[1] $A \in \mathbb{R}^{m\times n}_{\geq 0}$, $b \in \mathbb{R}^n_{>0}$, and $c \in \mathbb{R}^m_{>0}$. Abusing notation, we also let (C) denote the optimal value of the LP (C). We encountered a covering LP previous when designing a randomized approximation algorithm for set cover.

**LP Duality** In today's discussion we are particularly related to the connection between (P) and (C). When $A, b, c$ refer to the same objects in both (P) and (C), then the **LP duality theorem** tells us the following.

$$(\mathsf{P}) = (\mathsf{C}). \tag{$\star$}$$

Today we will prove ($\star$) as well as explore an interesting application in network design in Section 3. To prove ($\star$), we can (of course) first interpret ($\star$) as a combination of two inequalities, "(P) $\leq$ (C)" and "(P) $\geq$ (C)". One of these is much easier to prove than the other. Let $x$ be a feasible solution to (P) and let $y$ be a feasible solution to (C). We have

$$\langle b, x\rangle \overset{\text{(a)}}{\leq} \left\langle A^T y, x\right\rangle = \langle y, Ax\rangle \overset{\text{(b)}}{\leq} \langle y, c\rangle$$

(a) is because $A^T y \leq b$ and $x \geq \mathbb{0}$. (b) is because $y \geq \mathbb{0}$ and $Ax \leq c$. If we take $x$ and $y$ to be optimum solutions for (P) and (C), respectively, then we obtain the inequality

$$(\mathsf{P}) \leq (\mathsf{C}).$$

The opposite direction, "(P) $\geq$ (C)", is not as simple. We will return to proving it after a few motivating examples.

[1] Of course, in (C), we could have written $A$ instead of its transpose $A^T$, and swapped $b$ and $c$, which would more closely resemble (P). It is convenient for the subsequent discussion on LP duality for $A$, $b$ and $c$ to have the same dimensions in (P) and (C).

## 1.1 Examples

### 1.1.1 Set cover and packing

### 1.1.2 Max-flow min-cut

$$\begin{aligned}
\text{maximize } & \lambda \text{ over } \lambda > 0 \text{ and } f : E \to \mathbb{R}_{\geq 0} \\
\text{s.t. } & f(e) \leq c(e) \text{ for all } e \in E, \\
& \sum_{e \in \partial^+(v)} f(e) - \sum_{e \in \partial^-(v)} f(e) = 0 \text{ for all } v \in V \setminus \{s, t\}, \\
& \sum_{e \in \partial^+(s)} f(e) - \sum_{e \in \partial^-(s)} f(e) = \lambda, \\
& \sum_{e \in \partial^-(t)} f(e) - \sum_{e \in \partial^+(t)} f(e) = \lambda.
\end{aligned} \tag{1}$$

Here $\partial^+(v)$ denotes the set of directed edges starting from $v$, and $\partial^-(v)$ denotes the set of directed edges ending at $v$.

The above formulation is compact, in the sense that it has only a polynomial number of constraints. Thus it can be solved by an LP solver in polynomial times. (There are also a number of fast combinatorial algorithms for maximum flow, several of which are included in introductory algorithms classes.)

Recall that any $(s,t)$-flow can be decomposed as a nonnegative combination of $s \to t$ paths. This allows us to write max flow as a **path packing LP** instead. Let $\mathcal{P}_{s,t}$ denote the set of paths from $s$ to $t$.

$$\text{maximize } \sum_{p \in \mathcal{P}_{s,t}} x_p \text{ over } x : \mathcal{P}_{s,t} \to \mathbb{R}_{\geq 0} \text{ s.t. } \sum_{p : e \in p} x_p \leq c_e \text{ for all } e \in E. \tag{2}$$

First, there are an exponential number of variables, with a variable $x_p$ for each path $p : s \rightsquigarrow t$. Second, compared to the "compact" formulation in (1) above, the path packing formulation is conceptually very simple. We are packing a maximum number of $s \rightsquigarrow t$ paths subject to cardinality constraints. It might seem odd that the exponentially large path packing formulation is conceptually cleaner then the polynomial size compact formulation, but this dynamic arises for many other problems.

For our discussion, we prefer (2) because we can directly apply LP-duality. "Packing points into sets" becomes "covering points into sets", "capacities" of sets become "costs", and the "profit" of each becomes a demand. For multicommodity flow, "points" are paths and "sets" are edges. Thus the dual of (2) is

$$\text{minimize } \sum_{e \in E} c(e) y_e \text{ over } y : E \to \mathbb{R}_{\geq 0} \text{ s.t. } \sum_{e \in p} y_e \geq 1 \text{ for all (s,t) paths } p \in \mathcal{P}_{s,t}.$$

### 1.1.3 Multicommodity flow and sparsest cut

Let $G = (V, E)$ be a graph with positive edge capacities $c : E \to \mathbb{R}_{\geq 0}$. A **multicommodity flow** consists of, for every pair $s, t \in V$, an $(s,t)$-flow $f_{s,t} : E \to \mathbb{R}_{>0}$. The "congestion" on an edge induced by a multicommodity flow $\{f_{s,t} : s, t \in V\}$ is the ratio of the total amount of flow on that edge to the capacity on the edge:

$$\text{congestion on } e = \frac{\sum_{e \in E} |f_{s,t}(e)|}{c(e)}.$$

Here we consider two problems for the special case of undirected graphs. In both cases, the input consists of an undirected capacitated graph $G = (V, E, c)$ and (symmetric) demands $d : \binom{V}{2} \to \mathbb{R}_{\geq 0}$.

**Concurrent flow.** In the **concurrent multicommodity flow** problem, the goal is to find a multicommodity flow routing $d$ of minimum congestion. Equivalent, the goal is to route $\lambda d$ without violating any edge capacities for $\lambda$ as large as possible. This value of $\lambda$ is called the *maximum concurrent flow*. Note that when $d$ only consists of one nonzero demand $d(s,t)$, this is equivalent to the maximum $\{s,t\}$-flow problem.

**Sparsest cut.** In the **sparset cut** problem, the goal is to partition the vertices into $(S, \bar{S})$ (where $\bar{S} = V \setminus S$) to minimize the ratio

$$\frac{\sum_{e \in \partial(S)} c(e)}{\sum_{s \in S, t \in \bar{S}} d(s,t)}.$$

Here $0/0$ is interpreted as $+\infty$, the worst possible score. Note that when $d$ only consists of one nonzero demand $d(s,t)$, this is equivalent to the minimum $\{s,t\}$-cut problem.

**LP duality** We might take $\{s,t\}$ max-flow min-cut theorem as an inspiration for a similar relation between concurrent flow and sparsest cut. Let us take an LP based approach, and first write down an LP for multicommodity flow. Following the lead of $\{s,t\}$-max flow, we will design a packing LP based on packing paths. For $s, t \in V$, let $\mathcal{P}_{s,t}$ be the set of paths from $s$ to $t$. Let $\mathcal{P} = \prod_{s,t \in V} \mathcal{P}_{s,t}$ be the product of these families of paths. Each object $p \in \mathcal{P}$ is a *bundle* of paths, consisting of one path $p_{s,t} \in \mathcal{P}_{s,t}$ for every pair $s, t \in V$. A path packing formulation for concurrent multicommodity flow can now be written as follows.

$$\text{maximize} \sum_{p \in \mathcal{P}} x_p \text{ over } x : \mathcal{P} \to \mathbb{R}_{\geq 0} \text{ s.t. } \sum_{p} \sum_{s,t : e \in p(s,t)} x_p \leq c(e)$$

for all $e \in E$. The *dual* of this LP is the following covering LP.

$$\text{minimize} \sum_{e} c(e) y_e \text{ over } y : E \to \mathbb{R}_{\geq 0} \text{ s.t. } \sum_{s,t \in V} d(s,t) \sum_{e \in p(s,t)} y_e \geq 1 \text{ for all } p \in \mathcal{P}.$$

To interpret this LP, observe that for every $p \in \mathcal{P}$ and $s, t \in V$, $\sum_{e \in p(s,t)} y_e$ is the length of the path $p(s,t)$ w/r/t the edge lengths $y_e$. For a fixed vector $y$, let $\delta_y : V \times V \to \mathbb{R}_{\geq 0}$ give the shortest path distances w/r/t $y$. Then the LP can be written as

$$\text{minimize} \sum_{e} c(e) y_e \text{ over } y : E \to \mathbb{R}_{\geq 0} \text{ s.t. } \sum_{s,t \in V} d(s,t) \delta_y(s,t) \geq 1.$$

Let us consider the special case of *undirected graphs*. In the LP above, $\delta_y$ gives a metric, and in any optimal solution, we have $\delta_y(s,t) = y_e$ for any edge $e = \{s,t\}$. This allows us to rewrite the LP (again!) as follows. For a metric $\delta : \binom{V}{2}$ and an edge $e = \{s,t\}$, let us denote $\delta(e) = \delta(s,t)$. Consider the following equivalent LP (for unidrected graphs).

$$\text{minimize} \sum_{e} c(e) \delta(e) \text{ over all metrics } \delta : E \to \mathbb{R}_{\geq 0} \text{ s.t. } \sum_{s,t \in V} d(s,t) \delta(s,t) \geq 1.$$

By scaling, this is equivalent to the following LP.

$$\text{minimize } \frac{\sum_e c(e)\delta(e)}{\sum_{s,t\in V} d(s,t)\delta(s,t)} \text{ over all metrics } \delta : \binom{V}{2} \to \mathbb{R}_{\geq 0}. \tag{3}$$

Finally, we recognize this last LP as a relaxation to the sparsest cut problem. Indeed, any cut $(S, \bar{S})$ gives rise to a (boring) metric $\delta_S$ defined by

$$\delta_S(s,t) = \begin{cases} 0 & \text{if } s,t \in S \\ 1 & \text{if } s \in S \text{ and } t \in \bar{S} \\ 1 & \text{if } s \in \bar{S} \text{ and } t \in S \\ 0 & \text{if } s,t \in \bar{S}. \end{cases}$$

Let us the value of the LP (3) the "minimum fractional sparsest cut". We now have the following.

$$(\text{max. concurrent flow}) \overset{(c)}{=} (\text{min. fractional sparsest cut}) \overset{(d)}{\leq} (\text{min. sparsest cut}).$$

Here (c) applies LP-duality ($\star$). (d) is because minimum fractional sparsest cut is a relaxation of minimum sparsest cut.

Recall that with $\{s,t\}$ max flow, the maximum flow equals the minimum cut. With concurrent flow, we are consider the multiple-source-sink generalization of $\{s,t\}$-max flow. In this sense, we have "max flow $\leq$ min cut". One might ask if also "min cut $\leq$ max flow", or more generally, for the smallest factor $\alpha \geq 1$ such that

$$(\text{min. sparsest cut}) \leq \alpha(\text{max. concurrent flow}).$$

This would give an "$\alpha$-approximate max-flow min-cut" theorem for multiple commodities. This will be the topic of our next discussion.

# 2 Algorithmic proof of LP Duality

To simplify notation, we rescale $(\mathsf{P})$ as follows. We first scale $b$ to $\mathbb{1}$ (replacing each $A_{ij}$ with $A_{ij}/b_j$). We also scale $c$ to $\mathbb{1}$ (replacing each $A_{ij}$ with $A_{ij}/c_i$). We are thus left with the following normalized packing problem,

$$\text{maximize } \langle \mathbb{1}, x\rangle \text{ over } x \in \mathbb{R}^n_{\geq 0} \text{ s.t. } Ax \leq \mathbb{1} \tag{$\mathsf{P}_{\mathbb{1}}$}$$

and the normalized covering problem,

$$\text{minimize } \langle \mathbb{1}, y\rangle \text{ over } y \in \mathbb{R}^m_{\geq 0} \text{ s.t. } A^T y \geq \mathbb{1}. \tag{$\mathsf{C}_{\mathbb{1}}$}$$

We note that the rescaling did not affect the optimum value of either LP: $(\mathsf{P}) = (\mathsf{P}_{\mathbb{1}})$, and $(\mathsf{C}) = (\mathsf{C}_{\mathbb{1}})$.

## 2.1 The partition function

To prove LP-duality, we will use a function $\pi :^m \to \mathbb{R}$, alternatively called the **partition** or **soft-max** function, defined by

$$\pi(x) = \log\left(\sum_i e^{x_i}\right).$$

**Lemma 1.** *For all $x \in \mathbb{R}^m$,*

$$x \leq \pi(x) \leq x + \log(m).$$

**Lemma 2.** *For all $x$, $\pi'(x)$ is positive and its coordinates sum to 1.*

## 2.2 A continuous algorithm

Let $\epsilon > 0$ be given, and let $\eta = \log(m)/\epsilon$. Consider the function $f : \mathbb{R}^n_{\geq 0} \to \mathbb{R}$ defined by

$$f(x) \overset{\text{def}}{=} A^T \pi'(\eta A x). \tag{4}$$

We think of $f$ is a continuous approximation to $\max_i (Ax)_i$, as follows.

**Lemma 3.** *For all $x$,*

$$\max_i (Ax)_i \leq f(x) \leq \max_i (Ax)_i + \epsilon.$$

Note that the derivative of $f$,

$$f'(x) = \frac{1}{\eta} \pi(\eta A x),$$

can be interpretted as a convex combination of the original packing constraints.

**Lemma 4.** *For any $x \in \mathbb{R}^n$, there is a coordinate $j \in [n]$ such that*

$$f'_j(x) \leq \frac{1}{(\mathsf{C})}.$$

*Proof.* Suppose by contradiction that

$$f'(x) = \frac{1}{\eta} \pi(\eta A x) \geq \frac{1+\epsilon}{(\mathsf{C})} \mathbb{1}$$

for some $\epsilon > 0$. Then

$$y = \frac{(\mathsf{C})}{1+\epsilon} \pi'(\eta A x)$$

is a feasible solution to the covering LP with objective value

$$\langle \mathbb{1}, y \rangle = \frac{(\mathsf{C})}{1+\epsilon} < (\mathsf{C}),$$

a contradiction to $(\mathsf{C})$ being the optimum value. ■

**Theorem 5.** $(\mathsf{P}) = (\mathsf{C})$.

*Proof.* We will show that for all $\epsilon > 0$, $(\mathsf{P}_{\mathbb{1}}) \geq 1/1 + \epsilon$. Then taking $\epsilon \downarrow 0$ gives the desired result.

Let $\epsilon > 0$, and consider the function $f(x)$ defined in (4) above. We define a *continuous path of solutions* $x(t) \in \mathbb{R}^n_{\geq 0}$, for $t \in [0, 1-\epsilon]$. Initially, we set $x(0) = \mathbb{0}^n$. For each instance of time $t$, by Lemma 4, there is a coordinate $i(t) \in [n]$ such that $f'_i(x(t)) \leq 1$. We define the *rate of change* of $x(t)$ as

$$x'(t) = e_{i(t)}.$$

That is, at each time step $t$, we increase the coordinate $i(t)$ of $x$ such that $f'_i(x(t)) \leq 1$. We claim that $x(1)/1+\epsilon$ is a feasible solution with objective value $\geq 1/(1+\epsilon)$. At $t = 1$, the objective value is

$$\langle \mathbb{1}, x(1-\epsilon) \rangle \overset{\text{(a)}}{=} \int_0^1 \frac{d}{dt} \langle \mathbb{1}, x(t) \rangle \, dt = \int_0^1 \langle \mathbb{1}, x'(t) \rangle \, dt \overset{\text{(b)}}{=} \int_0^1 1 \, dt = 1.$$

Here (a) is by the fundamental theorem of calculus (noting that $\langle \mathbb{1}, x(0)\rangle = \langle \mathbb{1}, \mathbb{0}\rangle = 0$). (b) is because $x'(t)$ is the indicator function $e_{i(t)}$. It remains to show that $x(1)/(1+\epsilon)$ is feasible. To analyze the value of $f(x(1))$, we have

$$\begin{aligned} f(x(1)) - f(x(0)) &\overset{\text{(c)}}{=} \int_0^1 \frac{d}{dt} f(x(t))\, dt \overset{\text{(d)}}{=} \int_0^1 \langle f'(x(t)), x'(t)\rangle\, dt \\ &\overset{\text{(e)}}{=} \int_0^1 \left(f'(x(t))\right)_{i(t)} dt \overset{\text{(f)}}{\leq} \int_0^1 1\, dt = 1. \end{aligned}$$

Thus,

$$\max_i (Ax(1))_i \overset{\text{(g)}}{\leq} f(x(1)) \overset{\text{(h)}}{\leq} 1 + f(x(0)) = 1 + \epsilon.$$

Thus $x(1)/(1+\epsilon)$ is a feasible solution with objective value $\geq 1/(1+\epsilon)$. ■

### 2.3 A discrete algorithm

**Lemma 6.** *Let $x, y \in \mathbb{R}^n$ and $\epsilon > 0$, and suppose $x_i \leq y_i \leq x_i + \epsilon$ for all coordinates $i$. Then for all $i$,*

$$e^{-\epsilon}\pi_i'(x) \leq \pi'(y) \leq e^{\epsilon}\pi_i'(x).$$

*Proof.* Recall that

$$\pi_i'(x) = \frac{e^{x_i}}{\sum_j e^{x_j}}.$$

If $x_i \leq y_i \leq x_i + \epsilon$, then

$$e^{x_i} \leq e^{y_i} \leq e^{\epsilon} e^{x_i}$$

for all $i$. The claim follows. ■

**Theorem 7.** *Let $\epsilon > 0$ and consider an LP of the form ($\mathsf{P}_{\mathbb{1}}$). Suppose one has access to an oracle that, given any distribution $p \in \Delta_m$, one can compute a $(1+\epsilon)$-approximation to the Lagrangian relaxation*

$$\textit{maximimize } \langle \mathbb{1}, x\rangle \textit{ s.t. } \langle p, Ax\rangle \leq 1$$

*in polynomial time. Then in $O\big(m\log(n)/\epsilon^2\big)$, one can compute a $(1 - c\epsilon)$-approximate feasible solution to ($\mathsf{P}_{\mathbb{1}}$) for a (small) universal constant $c \geq 1$.*

*Proof.* The algorithm is based on discretizing the continuous proof in Section 2.2. Recall that each time step $t \in [0,1]$, we had a partial solution $x(t) \in \mathbb{R}^n_{\geq 0}$, and in each time step $t$, we choose a coordinate $i(t)$ such that $f'_{i(t)}(x(t)) \leq 1$. We discretize this as follows.

Each iteration $k$, we have a solution $x^k$ at time $t^k$. In the first iteration, we have $x^1 = \mathbb{0}$ and $t^1 = 1$. We approximately solve the Lagrangian relaxation and obtain a point $z_k$ such that

$$\left\langle \mathbb{1}, z^k \right\rangle \geq (1-\epsilon)\,\text{OPT} \;\text{ and }\; \left\langle f'\left(x^k\right), Az^k \right\rangle \leq 1.$$

Let $\delta^k$ large as possible such that $t^k + \delta^k \leq 1$ and

$$\delta^k \eta A z^k \leq \frac{\epsilon}{2}\mathbb{1}.$$

Observe that by Lemma **??**, because $\epsilon\eta A x^k \leq \epsilon\eta x^{k+1} \leq \epsilon\eta A x^k + \frac{\epsilon}{2}\mathbb{1}$, we have

$$f'\left(x^{k+1}\right) \leq e^{\epsilon} f'\left(x^k\right).$$

Now, imagine $x(t)$ as a continuous solution we

$$x(t^k) = x^k$$

for all $k$ and we interpolate linearly inbetween. For all $t \in (t^k, t^{k+1})$, we have

$$x'(t) = z^k.$$

In particular, for $t \in (t^k, t^{k+1})$, we have

$$\left\langle x'(t), \mathbb{1}\right\rangle \geq 1 - \epsilon$$

and

$$\left\langle f'(x(t)), Ax'(t)\right\rangle = \left\langle f'(x(t)), Az^k\right\rangle \leq (1+\epsilon)\left\langle f'\left(x^k\right), Az^k\right\rangle \leq 1 + \epsilon.$$

Repeating the calculations from Theorem 5, we see that

$$\langle \mathbb{1}, x(1)\rangle \geq 1 - \epsilon$$

and

$$Ax(1) \leq e^{\epsilon}\mathbb{1} + \epsilon \leq (1 + 3\epsilon)\mathbb{1}.$$

Scaling down $x(1)$ gives the desired soluton. ■

An alternative form or Theorem 7 is as follows.

**Theorem 8.** *Consider an LP of the form* $(\mathsf{P}_{\mathbb{1}})$. *Suppose that for any distribution* $p \in \Delta_m$ *over, one can find a feasible solution* $x$ *of value* $\geq \alpha$ *to the Lagrangian relaxation*

$$\text{maximimize } \langle \mathbb{1}, x\rangle \text{ s.t. } \langle p, Ax\rangle \leq 1$$

*in polynomial time. Then in* $O\left(m \log(n)/\epsilon^2\right)$ *iterations, where each iteration produces a feasible point* $x$ *of value* $\alpha$ *to a relaxation of the above form, one can compute a point* $x \in \mathbb{R}^n_{\geq 0}$ *such that*

$$\langle \mathbb{1}, x\rangle \geq (1-\epsilon)\alpha \text{ and } Ax \leq \mathbb{1}.$$

## 3 Oblivious routing

Recall the concurrent multicommodity flow problem, where given a graph $G = (V, E)$ with edge capacities $c : E \to \mathbb{R}_{\geq 0}$ and demands $d : V \times V \to \mathbb{R}_{>0}$, the goal is to simulatenously routine $d(s,t)$ units of flow from $s$ to $t$ for all $s,t \in V$, with minimum congestion. Here we consider the multicommodity flow problems for *undirected* graphs.

An **oblivious routing scheme** consists of a unit $\{s,t\}$-flow $f_{s,t}$ for every $s,t \in V$. Given demands $d : \binom{V}{2} \to \mathbb{R}_{\geq 0}$ and an oblivious routing scheme $\mathcal{F} = \{f_{s,t} : s,t \in V\}$, we can route $d(s,t)$ units of flow along $f_{s,t}$ for all $s,t \in V$ to obtain a multicommodity flow. For a parameter $\alpha \geq 1$, we say that $\mathcal{F} = \{f_{s,t} : s,t \in V\}$ is an ***α*-competitive routing scheme** if for any set of demands $d : \binom{V}{2} \to \mathbb{R}_{\geq 0}$, the multicommodity flow $\{d(s,t)f_{s,t} : s,t \in V\}$ has congestion within an $\alpha$-approximate factor of OPT. That is, $\mathcal{F}$ gives an $\alpha$-approximate solution for *any* set of demands $d$. Our goal is to construct an $\alpha$-competitive routing scheme. Note that the minimum congestion $s \to t$ flow for all $s$ and $t$ gives a $\binom{n}{2}$-competitive routing scheme. *A priori*, there is no reason why anything better should be possible.

We will develop routing schemes of the following simple type. A **routing tree** is a routing scheme obtained by embedding a tree with leaves corresponding to $V$ into $G$. More precisely, a routing tree consists of an auxiliary tree $T$ where each vertex in the tree corresponds to a node in the graph and each edge in the tree corresponds to a path in the graph with the following properties:

1. If $e = \{u,v\} \in T$ is an edge in the tree, then $e$ corresponds to a path between the vertices corresponding to $u$ and $v$ in $G$.

2. The leaves of $T$ are in one-to-one correspondance with the vertices $V$.

A routing tree implies the following routing scheme. For every pair of vertices $s,t \in V$, there are two leaves in $T$ corresponding to $s$ and $t$. The unique path in $T$ between these leaves implies a walk from $s$ to $t$ in $G$, where we concatenate the paths in $G$ corresponding to the edges in $T$ along the $s$-leaf to $t$-leaf path in $T$.

**Theorem 9** (Räcke (2008))**.** *In randomized polynomial time, one can compute a convex combination of routing trees that form a $O(\log n)$-competitive oblivious routing scheme.*

For a pair of vertices $s,t \in V$, let $T_{s,t}$ be the $\{s,t\}$-walk in $G$ given by $T$. For any edge $e = \{u,v\} \in E$, we denote $T_e = T_{s,t}$. For a walk $T_{s,t}$ and an edge $e \in E$, let $T_{s,t}(e)$ be the number of times $e$ appears in the walk $T_{s,t}$. Let $\mathcal{T}$ denote the family of all routing trees in $G$. Consider the following LP.

$$\text{maximize} \sum_{T \in \mathcal{T}} x_T \text{ over } x : \mathcal{T} \to \mathbb{R}_{\geq 0} \text{ s.t. } \sum_{T \in \mathcal{T}} \sum_{d \in E} c(d) T_d(e) \leq c(e) \text{ for all } e \in E. \qquad (\mathsf{P})$$

This LP relates to oblivious routing as follows.

**Lemma 10.** *Suppose $x : \mathcal{T} \to \mathbb{R}_{\geq 0}$ is a feasible solution to* (P)*, packing $\alpha$ fractional routing trees into $G$. Then $x/\alpha$ is a convex combination of routing trees that give a $(1/\alpha)$-competitive oblivious routing scheme.*

*Proof.* Suppose we simulatenously route, for every routing tree $T \in \mathcal{T}$ and every edge $e = \{s,t\} \in E$, $x_T c(e)$ units of flow from $s$ to $t$ along the walk $T(s,t)$. Because $x$ is a feasible solution to (P), this routes a total of $\alpha c(e)$ units of flow from $s$ to $t$ for each edge $e = \{s,t\} \in E$ with congestion 1. By scaling, for any $\gamma > 0$, we can simultaenously route $\gamma c(e)$ units of flow from $s$ to $t$ for every edge $e = \{s,t\}$ with congestion $\gamma/\alpha$.

Now, given any demands $d : \binom{V}{2} \to \mathbb{R}_{\geq 0}$ that can be routed in $G$ with congestion $\gamma$, consider the multicommodity flow where we route, for every $s, t \in V$ and $T \in \mathcal{T}$, $(x_T/\alpha)d(s,t)$ units of flow from $s$ to $t$ along the walk $T(s,t)$. Since the demands $d$ can be routed with congestion $\gamma$, the congestion induced by the fractional tree routing is no worse then the congestion induced by routing $\gamma c(e)$ units of flow from $s$ to $t$ through the trees for every edge $e = \{s,t\} \in E$. By the preceding discussion, this gives congestion at most $\gamma/\alpha$. ■

Given Lemma 10, it remains to try to find a fractional solution $x$ to the LP (P) with as large an optimal value as possible. We will attempt to solve the LP using the algorithm from Section 2.3. To this end, it is convenient to normalize the LP as follows.

$$\text{maximize} \sum_{T \in \mathcal{T}} x_T \text{ over } x : \mathcal{T} \to \mathbb{R}_{\geq 0} \text{ s.t. } \frac{1}{c(e)} \sum_{T \in \mathcal{T}} \sum_{d \in E} c(d) T_d(e) \leq 1 \text{ for all } e \in E. \tag{$\mathsf{P}_1$}$$

Recall that the algorithm in Section 2.3 is based on solving a sequence of Lagrangian relaxations, that form packing problems with only a single packing constraint. Given a probability distribution $p \in \Delta_m$ over $E$, the Lagrangian relaxation of (**??**) has the form

$$\text{maximize} \sum_{T \in \mathcal{T}} x_T \text{ over } x : \mathcal{T} \to \mathbb{R}_{\geq 0} \text{ s.t. } \sum_{T \in \mathcal{T}} \sum_{e \in E} \frac{p_e}{c(e)} \sum_{d \in E} c(d) T_d(e)$$

This is a fractional packing problem where all items have the same value and there is only one packing constraint, which can be interpretted as a cost, as follows.

$$\operatorname{cost}(T \,|\, p) = \sum_{e \in E} \frac{p_e}{c(e)} \sum_{d \in E} c(d) T_d(e).$$

A tree $T$ with cost $\operatorname{cost}(T \,|\, p)$ gives a solution of value $1/\operatorname{cost}(T \,|\, p)$ to the relaxation. For any fixed $\beta > 0$, if we can consistently find a tree $T$ with cost $< \beta$ for any distribution $p$, then by Theorem 7, we can obtain a $(1+\epsilon)\beta$-approximation to (P) for any fixed $\epsilon > 0$.

Towards this end, the first lemma reinterprets $\operatorname{cost}(T \,|\, p)$ in terms of the distances of its walks.

**Lemma 11.** *Let $p \in \Delta_E$ be a probability distribution over $E$. Consider the shortest path metric on $G$ w/r/t edge lengths $\frac{p_e}{c(e)}$. For a routing tree $T$, let $d_T : V \times V \to \mathbb{R}_{\geq 0}$ be the metric where $d_T(u,v)$ is the length of the walk $T_{u,v}$ w/r/t the edge lengths $p_e/c(e)$. Then*

$$\operatorname{cost}(T \,|\, p) = \sum_{f \in E} c(f) d_T(f).$$

*Proof.* We have

$$\operatorname{cost}(T) = \sum_{e \in E} \frac{p_e}{c(e)} \sum_{f \in E} c(f) T_f(e) \overset{\text{(a)}}{=} \sum_{f \in E} c(f) \sum_{e \in E} \frac{p_e}{c(e)} T_f(e) \overset{\text{(b)}}{=} \sum_{f \in E} c(f) d_T(f).$$

(a) interchanges sums. For (b), we recall that $T_f(e)$, where $f = \{s,t\}$, is the number of times edge $e$ appears in the walk $T_{s,t}$ from $s$ to $t$. Thus $\sum_{e \in E} \frac{p_e}{c(e)} T_f(e) = d_T(f)$ for each edge $f$. ■

**Lemma 12.** *Let $p \in \Delta_E$ be a probability distribution over $E$. In randomized polynomial time, one can compute a routing tree $T$ such that*

$$\operatorname{cost}(T) \leq O(\log n).$$

*Proof.* We apply the randomized tree metric algorithm from our previous discussion with respect to the shortest path metric induced by edge lengths $p_f/c(f)$. This produces a randomized routing tree $T$ such that

$$\mathrm{E}[d_T(f)] \leq \frac{p_f}{c(f)}$$

for all edges $f \in E$. By Lemma 11, we have

$$\mathrm{E}[\mathrm{cost}(T)] = \sum_{f \in E} c(f)\,\mathrm{E}[d_T(f)] \leq O(\log n) \sum_{f \in E} p_f = O(\log n),$$

as desired. ■

To prove Theorem 9, we apply the discrete LP algorithm from Section 2.3 to the packing LP $(\mathsf{P}_1)$ with a constant value of $\epsilon$. For every Lagrangian relaxation induced by $p \in \Delta_E$, by Lemma 12, we can produce a fractional solution of value $\geq \Omega(1/\log n)$. Thus we obtain feasible solution to $(\mathsf{P}_1)$ of value $\Omega(1/\log n)$. This in turn implies a $O(\log n)$-competitive oblivious routing scheme.

## References

Räcke, Harald (2008). “Optimal hierarchical decompositions for congestion minimization in networks”. In: *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*. Ed. by Cynthia Dwork. ACM, pp. 255–264.