A Simpler and Self-Contained Analysis for Approximate Packing and Covering LPs via Weight Update Rule

Sorrachai Yingchareonthawornchai
Department of Computer Science and Engineering, Michigan State University, USA
yingchar@cse.msu.edu

Abstract

Algorithms for fast approximate solving of packing and covering LPs have been presented in various forms, e.g. by Plotkin-Shmoys-Tardos and Garg-Könemann, which are later simplified through the lens of multiplicative weight update (MWU) by Arora-Hazan-Kale. We can think of MWU framework as a reduction from solving a packing LP with many constraints, to solving a simpler LP with only one (average) constraint. The beauty of MWU framework is that not only MWU-style algorithm can solve the approximate primal LP, but also can solve the approximate dual LP at the same time. This fact is useful for efficiently solving an LP with exponential number of constraints, but its dual has only linear number of constraints. A natural question is that is there a natural MWU algorithm that solves both primal and dual LPs simultaneously with elementary analysis? In this work, we present a MWU-style algorithm based on Garg-Könemann that admits a simpler analysis, allowing to obtain a stronger result. The key novelty is in the analysis where we directly bound primal and dual gaps over time whereas Garg-Könemann and Arora-Hazan-Kale focus only on obtaining the primal solutions. Our result can be viewed as another refinement of Garg-Könemann and Arora-Hazan-Kale with natural choices of parameters and oracle.

2012 ACM Subject Classification Design and Analysis of Algorithms

Keywords and phrases MWU, Linear Programming, Algorithm Design

Digital Object Identifier 10.4230/OASIcs...1

1 Introduction

Algorithms for fast approximate solving of packing and covering LPs have been presented in various forms, e.g., by Plotkin-Shmoys-Tardos (PST) [11], and Garg-Könemann (GK) [8], which are later simplified through the lens of multiplicative weights update (MWU) by Arora-Hazan-Kale (AHK) [2]. PST [11] give early MWU-style algorithms, but their running time depends on width parameter of the LPs. The width is considered the main bottleneck of the running time since it can be huge. Later on, GK [8] present a clever technique to obtain width-independent algorithms for multicommodity flow and packing LPs. Recently, AHK [2] popularize MWU algorithms in their comprehensive survey which we can view PST and GK through MWU framework. The framework follows an abstract problem called learning from experts' advice problem, which is applicable in wide range of problems including solving approximate packing LPs. Packing LP is a linear program of the form $\max\{\vec{v}\cdot\vec{x} : A\vec{x} \leq \vec{c}, \vec{x} \geq 0\}$ and its dual covering LP of the form $\min\{\vec{c}\cdot\vec{y} : A^T\vec{y} \geq \vec{v}, \vec{y} \geq 0\}$ where $A \in \mathbb{R}^{mn}_{\geq 0}, \vec{v} \in \mathbb{R}^n_{\geq 0}, \vec{c} \in \mathbb{R}^m_{\geq 0}$.

We can think of MWU framework as a reduction from solving an LP with m constraints, to solving a simpler LP with only one (average) constraint. We call the simpler LP as an oracle problem. In particular, packing LPs can be viewed as packing objects (such as trees,

Table 1 Comparisons of width-independent MWU-based algorithms (restricted to packing or covering LPs).

Algorithms	Primal	Dual	Simple[1]
GK[8]	Yes	No	No
AHK[2]	Yes	No	Yes
CJV[4]	Yes	Yes	No
This paper	Yes	Yes	Yes

paths, cuts, and so on) as much as we can without violating m constraints. The oracle problem is typically computing the minimum objects (minimum spanning trees, shortest paths, minimum cuts, and so on) with respect to current weights on each constraint. Here, a MWU-based algorithm starts by initializing weights of each constraint. Then, it calls an oracle to get the minimum weighted object. In turn, it penalizes every constraint that belongs to this minimum object by roughly a $1 + \epsilon$ multiplicative factor, for some small constant ϵ. We repeat this process until some stopping conditions. The running time is measured in terms of number of oracle calls. MWU algorithms differ on initialization, weight update rules, stopping condition, and oracle. For width-independent algorithm, GK show that $O(\frac{1}{\epsilon^2} m \log m)$ oracle calls suffices.

The beauty of MWU framework is that not only MWU-style algorithm can solve the approximate primal LP, but also can solve the approximate dual LP at the same time. It is folklore that we can use weights as dual variables; with appropriate scaling, we can show that primal and dual solutions converge towards to optimal solutions. This fact, however, does not immediately follow from algorithms by GK, and AHK since their parameters seem to engineer towards obtaining the approximate primal solution (more detail later in this section). Recently, Chekuri-Jayram-Vondrak (CJV) [4, 3] has sketched the proof of how to obtain such solutions. However, the full proof is not self-contained: to understand its full proof, one needs to consult [2] for standard MWU framework, and [4] for their proposed MWU algorithm, and [5] for its specialization to packing LPs. We note that their remarkable algorithm in [4] is based on discretization of continuous optimization via MWU framework, which is designed for solving more general problems than LPs, and is quite involved.

The fact that we can obtain dual solutions from the algorithm that solves primal LP gives us a technique to efficiently solve an LP with exponential number of constraints, but its dual has only linear number of constraints. For example, a recent breakthrough on fast LP solver for Held-Karp bounds for metric TSP [3] leverages this technique. It reduces to the problem of finding the minimum fractional 2-edge connected spanning subgraphs (2ECSS). However, 2ECSS LP has exponential number of constraints (i.e., every cut has at least two crossing edges). The dual LP of 2ECSS is a fractional cut-packing problem where there are only m constraints on each edge. In this way, we can use MWU framework on cut-packing problems and obtain 2ECSS solutions indirectly so that the number of oracle calls is still near-linear in number of edges.

A natural question is that is there a natural MWU algorithm that solves both primal and dual LPs simultaneously with elementary analysis? In this work, we present a MWU-style algorithm based on GK that admits a simpler analysis, allowing to obtain a stronger result. The key difference is in the analysis where we directly bound primal and dual gaps over time whereas GK and AHK focus only on obtaining the primal solutions. Our result can be viewed as another refinement of GK and AHK with natural choices of parameters and oracle. For concreteness, we will discuss on solving maximum flow LPs which we will formalize in

next Section. The oracle problem is thus the shortest path problem. Recall that MWU-based framework needs to specify (1) initial weights, (2) weight update rule (3) stopping condition and (4) oracle problem. Weight update rule and stopping conditions are similar for MWU framework. We will focus on difference in initial weights and oracles. The choices of parameters in GK, and AHK seem to be engineered toward obtaining the primal solutions in their proofs. In GK, they initialize weight to be $(1+\epsilon).((1+\epsilon)m)^{-1/\epsilon}$ where $\epsilon > 0$. In AHK, they initialize weights to be uniformly 1, and use oracle as a shortest path whose weight is normalized by capacity. This oracle allows them to relate analysis to the online learning from experts problem, which is an abstract problem for MWU framework. Our paper uses initial weight to be reciprocal to capacity of an edge. Intuitively, we should not prefer edge with high capacity as part of the shortest path. Our oracle is a pure shortest path of the current weights. We summarize in Table 1. We note that our simplicity assessment is based on whether or not the algorithm is natural and self-contained for readers who are new to this area.

Other related work. We focus on simplifying the classic result by Garg-Könemann [8] where we focus on solving an implicit LP with access to oracle where we count only number of calls to the oracle to get a near-optimal solution. Subsequent improvements can be made such as those in [7]. There are algorithms that solve more general LP such as positive LPs[9, 1, 6] in which algorithms tend to be more involved and far developed. For pure running time, it is possible to speedup the solving time even further when combined with dynamic algorithms. The key idea is we do not have to recompute everything from scratch. For example, for maximum multicommodity flow problem, it is possible to speed-up shortest path oracle by casting to decremental dynamic All-Pair-Shortest-Paths (APSP) problem [10]. This improves from $\tilde{O}(m^2)$ if we call shortest paths oracle repeatedly to $\tilde{O}(mn)$ with dynamic APSP where m is number of edges and n is number of nodes. Another example includes solving 2ECSS in nearly linear time using decremental minimum cuts [3].

The paper is organized as follows. We present an algorithm for solving approximate maximum flow and minimum cut in Section 2. Then, we generalize to solving general packing and covering LPs in Section 3. Finally, we conclude in Section 4.

2 Approximate Maximum Flow and Minimum Cut

We describe an algorithm for approximate maximum flow and minimum cut to demonstrate the main idea for the new analysis. Our analysis extends naturally to the general case of packing and covering LPs. We first describe the maximum flow problem in terms of packing LPs (with possibly exponential number of variables in the primal) as follows. In a flow network $G = (V, E)$ with edge capacity c_e for $e \in E$, let $\mathcal{P}$ be the set of all possible paths from a fixed source s to a fixed sink t. Our goal is to route the maximum possible amount of flow though paths in $\mathcal{P}$ such that total amount of flow on each edge does not exceed its capacity, $f(e) = \sum_{p \ni e} f_p \leq c_e$. The dual version is the minimum cut problem that is to find a minimum fractional weighted cut such that every path has length at least 1, $w(p) = \sum_{e \in p} w_e \geq 1$. We summarize the problems in Figure 1. We use n to denote number of vertices and m to denote number of edges of the graph.

Our algorithm is based on the MWU-style update rules.

2.1 Algorithm

Denote by $w^{(t)}$ the weight function at time t. We set initial weight of each edge to be $w_e^{(0)} = \frac{1}{c_e}$. We repeat the followings. At round t, we compute the shortest path $p^{(t)}$

$$\begin{aligned} \text{maximize} \quad & \sum_{p\in\mathcal{P}} f_p \\ \text{s.t.} \quad & f(e) \le c_e, \quad e \in E \\ & f_p \ge 0, \quad p \in \mathcal{P} \end{aligned} \qquad \begin{aligned} \text{minimize} \quad & \sum_{e\in E} c_e w_e \\ \text{s.t.} \quad & w(p) \ge 1, \quad p \in \mathcal{P} \\ & w_e \ge 0, \quad e \in E \end{aligned}$$

Figure 1 Maximum flow LP and its dual.

in the graph with current weight function $w^{(t)}$. Let e^* be the minimum capacity edge in path $p^{(t)}$. We route flow of value $f^{(t)} = c_{e^*}$ along this path. We update weights of edges along the path proportionally to their capacity: for each edge $e \in p^{(t)}$, we update $w_e^{(t+1)} = w_e^{(t)}(1+\frac{\epsilon f^{(t)}}{c_e})$. We stop this procedure when there is an edge with "high congestion". More precisely, let $\text{Flow}(e,t)$ be the amount of flow that edge e receives at round t. We define congestion $\mathsf{cong}^{(t)}(e) = \frac{\sum_{t=1}^{T}\text{Flow}(e,t)}{c_e}$ and $\mathsf{cong}^{(t)}(G) = \max_e \mathsf{cong}^{(t)}(e)$. We stop when $\mathsf{cong}^{(t)}(G) > \frac{\ln m}{\epsilon^2}$. This completes the description of the algorithm.

It is easy to see that $\mathbf{f}^{(t)}(e) = \frac{\sum_{j=1}^{t} Flow(e,j)}{\mathsf{cong}^{(t)}(G)}$ is a feasible solution to the primal, and that $\mathbf{D^{(t)}} = \{w_e^{(t)}/w(p^{(t)})\}_{e\in E}$ is a feasible solution for the dual, with the (dual) objective value of $D(t) = W^{(t)}/w(p^{(t)})$ where $W^{(t)} = \sum_{e\in E} c_e w_e^{(t)}$.

Below, we will argue that the primal and dual solutions converge to each other, which would imply that both of them are near-optimal. Let T be the final iteration. We denote the value of the primal solution $F = \frac{1}{\mathsf{cong}^{(T)}(G)}\sum_{t=1}^{T} f^{(t)}$ as the value of total flow routed according to the primal solution $\mathbf{f}^{(T)}$. For dual solution, we denote D as the minimum over all $D(t)$ for all t.

▶ **Theorem 1.** *After $O(\frac{m}{\epsilon^2}\log m)$ rounds, primal and dual solutions converge to $\frac{F}{D} \ge 1-O(\epsilon)$.*

Tracking the dual objectives:

The key player in the analysis is the value of the dual objective $W^{(t)}$, which will be used as a potential function. Here, we explicitly calculate such values. Since only edges on the path $p^{(t)}$ get updated, the increase of dual objective function is $\epsilon f^{(t)} w(p^{(t)})$. Thus, the change in this term follows the rule: $W^{(t+1)} = W^{(t)}(1+\frac{\epsilon f^{(t)} w(p^{(t)})}{w^{(t)}})$. We summarize this change in Proposition 1.

▶ Proposition 1. The values of $W^{(t)}$ change according to the following rule:

$$W^{(t+1)} = W^{(t)}(1+\frac{\epsilon f^{(t)} w(p^{(t)})}{W^{(t)}})$$

2.2 Analysis

We describe an overview of the proof, which is reminiscent of the proof of multiplicative weight update algorithms for learning from experts' advice problem. We use dual objective $W^{(t)}$ as a potential function. We can bound the global change in $W^{(t)}$ using Proposition 1 which gives us the upper bound of $W^{(t)}$. On the other hand, we can bound the local change in weights of each edge given by update rule, and use the fact that $W^{(t)} \ge c_e w_e^{(t)}$ to get the

lower bound of $W^{(t)}$. Finally, we can relate the primal and dual solutions via the upper and lower bounds of the dual objective at the final round, $W^{(T)}$. We next formalize the proof.

We formulate the upper and lower bounds of $W^{(T)}$ into two Lemmas.

▶ **Lemma 2.** *After T iterations,*

$$W^{(T)} \leq m \exp(\sum_{t=1}^{T} \frac{\epsilon f^{(t)}}{D(t)})$$

Proof. By Proposition 1 and the fact that $D(t) = \frac{W^{(t)}}{w(p^{(t)})}$, we get $W^{(t+1)} = W^{(t)}(1 + \frac{\epsilon f^{(t)}}{D(t)})$. Since we initialize weight of each edge e as $\frac{1}{c_e}$, we get $W^{(0)} = m$. We can expand the products to get $W^{(T)} = m \prod_{t=1}^{T}(1 + \epsilon \frac{f^{(t)}}{D(t)})$. The result follows by using the inequality $1 + x \leq e^x$. ◀

▶ **Lemma 3.** *After T iterations, and for all edge e,*

$$w_e^{(T)} \geq \frac{1}{c_e} \exp(\epsilon(1-\epsilon)\mathsf{cong}^{(T)}(e))$$

Proof. We first fixed an edge e. By update rule, $w_e^{(t+1)} = w_e^{(t)}(1+\epsilon\frac{f^{(t)}}{c_e})$ if e is in the shortest path, and no change otherwise. Recall that $\text{Flow}(e,t)$ is the amount of flow e received at round t which is either $f^{(t)}$ or 0. We can rewrite the local update as $w_e^{(t+1)} = w_e^{(t)}(1 + \epsilon\frac{\text{Flow}(e,t)}{c_e})$. Note that $w_e^{(0)} = \frac{1}{c_e}$ by initialization. We can expand the products to get the final weight at round T as $w_e^{(T)} = \frac{1}{c_e}\prod_{t=1}^{T}(1 + \epsilon\frac{\text{Flow}(e,t)}{c_e})$. The result follows by the inequality $1 + x \geq \exp(x - x^2)$ for $0 < x < 0.5$. We can write $w_e^{(T)} \geq \frac{1}{c_e}\exp(\sum_{t=1}^{T}(\epsilon\frac{\text{Flow}(e,t)}{c_e})(1 - \epsilon\frac{\text{Flow}(e,t)}{c_e})) \geq \frac{1}{c_e}\exp(\sum_{t=1}^{T}\epsilon\frac{\text{Flow}(e,t)}{c_e})(1-\epsilon)) = \frac{1}{c_e}\exp(\epsilon(1-\epsilon)\mathsf{cong}^{(T)}(e)$. ◀

We are ready to prove the main claim.

Proof of Theorem 1. At any iteration t, the dual objective function $W^{(t)}$ is at least $c_e w_e^{(t)}$. Using Lemma 2, and Lemma 3, we have:

$$\begin{aligned}
W^{(T)} &\geq c_e w_e^{(T)} \\
m\exp(\sum_{t=1}^{T}\frac{\epsilon f^{(t)}}{D(t)}) &\geq \exp(\epsilon(1-\epsilon)\mathsf{cong}^{(T)}(e)) \\
\frac{1}{\epsilon}\ln m + \sum_{t=1}^{T}\frac{f^{(t)}}{D(t)} &\geq (1-\epsilon)\mathsf{cong}^{(T)}(e) \\
\sum_{t=1}^{T}\frac{f^{(t)}}{D(t)} &\geq (1-\epsilon)\mathsf{cong}^{(T)}(e) - \frac{1}{\epsilon}\ln m
\end{aligned}$$

Next, we use the fact that D is the minimum dual solution so far. So, $\sum_{t=1}^{T}\frac{f^{(t)}}{D} \geq \sum_{t=1}^{T}\frac{f^{(t)}}{D(t)}$, and we get:

$$\sum_{t=1}^{T}\frac{f^{(t)}}{D} = \frac{1}{D}\sum_{t=1}^{T} f^{(t)} \geq (1-\epsilon)\mathsf{cong}^{(T)}(e) - \frac{1}{\epsilon}\ln m$$

Since this is true for any edge, it is true for the maximum congested edge, $\mathsf{cong}^{(T)}(G)$, So,

$$\frac{1}{D}\sum_{t=1}^{T} f^{(t)} \geq (1-\epsilon)\mathsf{cong}^{(T)}(G) - \frac{1}{\epsilon}\ln m$$
$$\frac{1}{D}\frac{\sum_{t=1}^{T} f^{t}}{\mathsf{cong}^{(T)}(G)} \geq (1-\epsilon) - \frac{\ln m}{\mathsf{cong}^{(T)}(G)\epsilon}$$
$$\frac{F}{D} \geq (1-\epsilon) - \frac{\ln m}{\mathsf{cong}^{(T)}(G)\epsilon}$$

Recall that our primal solution F is the total flow scaled down by the maximum congestion $\mathsf{cong}^{(T)}(G)$. Finally, the algorithm terminates when there is an edge with high congestion which means we stop when $\mathsf{cong}^{(T)}(G) \geq \frac{\ln m}{\epsilon^2}$. Hence,

$$\frac{F}{D} \geq (1-2\epsilon)$$

Finally, we show the number of iterations is at most $T = O(\frac{1}{\epsilon^2}\log m)$. This is because each iteration, there must be a bottleneck edge e, which has the lowest capacity in the shortest path. This means the congestion of e will increase by 1 since we route c_e amount of flow through this path. By our stopping condition, each edge can be the bottleneck at most $O(\frac{1}{\epsilon^2}\log m)$ times. Therefore, with m edges, total number of iterations is $O(\frac{m\log m}{\epsilon^2})$.

◀

3 Approximate Packing and Covering LPs

There is nothing special about maximum flow problem, which can be viewed a path packing problem. We will show that essentially the same proof goes through the general packing LPs. Packing and its dual covering LPs have the form as in Figure 2. Denote A as a matrix of size m by n. Let $\vec{f}, \vec{w}$ be primal and dual variables, respectively. Let $\vec{c}, \vec{v}$ be a vector of length m, and length n, respectively. All entries in $A, \vec{c}, \vec{v}$ are non-negative real numbers.

We follow a scheme from Garg-Könemann[8] that we can view primal LP as a variant of maximum flow as follows. We consider j-th column of A as the j-th path, shipping a flow of color j (e.g., gold) with value $v(j)$ per unit of flow. Also, we can view i-th row of A as the i-th edge, a pipe made by material i (e.g., wooden) with capacity $c(i)$. We have additional interaction rule: a unit of flow of color j consumes $A(i,j)$ unit of capacity of pipe i.

Notations. We will reuse and most of notations since the algorithm and analysis are similar to the maximum flow problem in previous section. We will slightly redefine them for LPs in Figure 2. As we shall see, many key Lemmas can be stated in the identical form as those in the previous section. For path p, we use $p \in \{1, 2, \ldots, n\}$ as an index for $\vec{v}$ such as $v(p)$. For edge e, we use $e \in \{1, 2, \ldots, m\}$ as an index for $\vec{w}$.

maximize	$\vec{v}\cdot\vec{f}$	minimize	$\vec{c}\cdot\vec{w}$
s.t.	$A\vec{f} \leq \vec{c}$	s.t.	$A^T\vec{w} \geq \vec{v}$
	$\vec{f} \geq 0$		$\vec{w} \geq 0$

Figure 2 General packing and cover LPs.

3.1 Algorithm

We set initial weight of each edge to be $\frac{1}{c_e}$. We repeat the followings. For current weights, we define the length of path $p \in \{1, 2, \ldots, n\}$ as $w(p) = \frac{1}{v(p)}(A^T \vec{w}^{(t)})_p = \frac{1}{v(p)} \sum_e A(e,p) w_e^{(t)}$ for $e \in \{1, 2, \ldots, m\}$. At round t, let $p^{(t)}$ be the shortest path whose length is $w(p^{(t)})$. For a fixed flow of color $p^{(t)}$, each edge e has normalized capacity of $\frac{c(e)}{A(e,p^{(t)})}$ unit. Let $e^{(t)}$ be the bottleneck edge with minimum normalized capacity. We route $\frac{c(e^{(t)})}{A(e^{(t)},p^{(t)})}$ unit of flow of color $p^{(t)}$. Note that flow value in this round is $f^{(t)} = \frac{v(p^{(t)})c(e^{(t)})}{A(e^{(t)},p^{(t)})}$. We update weights as follows: for each edge e, $w_e^{(t+1)} = w_e^{(t)}(1 + \epsilon \frac{c(e^{(t)})/A(e^{(t)},p^{(t)})}{c(e)/A(e,p^{(t)})})$. We repeat until there is an edge with high congestion. More precisely, define $\mathsf{cong}^{(t)}(e) = \frac{1}{c_e} \sum_p A(e,p) f_p$, and $\mathsf{cong}^{(t)}(G) = \max_e \mathsf{cong}^{(t)}(e)$. We stop when $\mathsf{cong}^{(t)}(G) > \frac{\ln m}{\epsilon^2}$. This completes the description of the algorithm.

Similarly to the case of maximum flow, we compute the change of the dual objective function, $W^{(t)} = \sum_e c_e w_e$.

▶ Proposition 2. The values of $W^{(t)}$ change according to the following rule:

$$W^{(t+1)} = W^{(t)}(1 + \epsilon \frac{f^{(t)} w(p^{(t)})}{W^{(t)}})$$

Proof. We can bound the change of dual objective function using local update as follows:

$$W^{(t+1)} = \sum_e c_e w_e^{(t+1)} = \sum_e c_e w_e^{(t)} + \epsilon \frac{c(e^{(t)})}{A(e^{(t)},p^{(t)})} \sum_e A(e,p^{(t)}) w_e^{(t)}$$

Recall that $f^{(t)} = v(p^{(t)}) \frac{c(e^{(t)})}{A(e^{(t)},p^{(t)})}$. So, we get:

$$W^{(t+1)} = W^{(t)} + \epsilon f^{(t)} \frac{1}{v(p^{(t)})} \sum_e A(e,p^{(t)}) w_e^{(t)} = W^{(t)} + \epsilon f^{(t)} w(p^{(t)})$$

The last equality follows since $\frac{1}{v(p^{(t)})} \sum_e A(e,p^{(t)}) w_e^{(t)} = w(p^{(t)})$ by definition of path length . ◀

3.2 Analysis

We can obtain primal and dual solutions in the similar way as in the case of maximum-flow. Let C be the maximum congestion over all edges at the end of the algorithm, $C = \mathsf{cong}^{(T)}(G) = \max_e \mathsf{cong}^{(T)}(e)$. The primal solution, F, is the total value of flow (of various colors) scaled down by a factor of maximum congestion, $F = \sum_{t=1}^{T} \frac{f^{(t)}}{C}$. For dual solution, define $w(p^{(t)})$ is the length of the shortest path at time t, and $W^{(t)}$ is the objective dual solution at the t. Each iteration t, we have a dual solution $D(t) = \frac{W^{(t)}}{w(p^{(t)})}$. We choose the dual solution D as the minimum $D(t)$ over all t.

▶ Proposition 3. Primal solution F and dual solution D are both feasible.

Proof. First we show that primal solution F is feasible. Consider when the algorithm terminates at time T. Let e^* be the edge (row of A) that is maximally congested. By definition of congestion, $(A\vec{f})_{e^*} = c(e^*)\mathsf{cong}^{(T)}(e^*)$. In other words, for all edge e, $(A\vec{f})_e = c(e)\mathsf{cong}^{(T)}(e) \leq c(e)\mathsf{cong}^{(T)}(e^*)$. Hence, by dividing $\vec{f}$ by $C = \mathsf{cong}^{(T)}(e^*)$, which is the

maximum congestion on every edge, we ensure that $(A\vec{f'})_e \leq c(e)$ where $\vec{f'} = \frac{1}{C}\vec{f}$. Hence, F is feasible.

Next, we show that the dual solution D is feasible. That is, we need to show that $D(t) = \frac{W^{(t)}}{w(p^{(t)})}$ is feasible for all t. Recall $p^{(t)}$ is the shortest path (row of A^T). By definition of path length, $\frac{1}{v(p^{(t)})}(A^T\vec{w})_{p^{(t)}} = w(p^{(t)})$. In other words, for all path p, $\frac{1}{v(p)}(A^T\vec{w})_p = w(p) \geq w(p^{(t)})$. Hence, by dividing by $w(p^{(t)})$, we ensure that $\frac{1}{v(p)}(A^T\vec{w'})_p \geq 1$ where $\vec{w'} = \frac{1}{w(p^{(t)})}\vec{w}$. Hence, D is feasible. ◀

▶ **Theorem 4.** *After $O(\frac{m}{\epsilon^2}\log m)$ iterations, the solutions $F/D \geq 1 - O(\epsilon)$.*

The proof is essentially the same as the case of maximum flow. The difference is in the details of how to get upper and lower bounds of the same potential function, $W^{(T)}$. We now present two key Lemmas.

▶ **Lemma 5.** *After T iterations,*

$$W^{(T)} \leq m\exp(\sum_{t=1}^{T}\frac{\epsilon f^{(t)}}{D(t)})$$

Proof. The proof is identical to Lemma 2 by using Proposition 2 and $D(t) = \frac{W^{(t)}}{w(p^{(t)})}$. ◀

▶ **Lemma 6.** *After T iterations, and for all edge e,*

$$w_e^{(T)} \geq \frac{1}{c_e}\exp(\epsilon(1-\epsilon)\mathsf{cong}^{(T)}(e))$$

Proof. By update rules, for each edge e,

$$w_e^{(t+1)} = w_e^{(t)}(1 + \epsilon\frac{c(e^{(t)})/A(e^{(t)}, p^{(t)})}{c(e)/A(e, p^{(t)})})$$

Fix an edge e. We can expand from T to the base case. So,

$$w_e^{(T)} = \frac{1}{c_e}\prod_{t=1}^{T}(1 + \epsilon\frac{c(e^{(t)})/A(e^{(t)}, p^{(t)})}{c(e)/A(e, p^{(t)})}) \geq \frac{1}{c_e}\exp(\epsilon(1-\epsilon)\frac{1}{c(e)}\sum_{t=1}^{T}A(e, p^{(t)})\frac{c(e^{(t)})}{A(e^{(t)}, p^{(t)})})$$

The inequality follows by $1 + x \geq e^{x-x^2}$. Note that $\frac{c(e^{(t)})}{A(e^{(t)}, p^{(t)})}$ represents amount of flow in day t of color $p^{(t)}$, and thus summing over all days we get total flow weighted by A which is the same as $c(e)\mathsf{cong}^{(T)}(e)$. Hence, $\sum_{t=1}^{T}A(e, p^{(t)})\frac{c(e^{(t)})}{A(e^{(t)}, p^{(t)})} = c(e)\mathsf{cong}^{(T)}(e)$. The result follows by plugging last equation into the previous inequality. ◀

Proof of Theorem 4. The proof of approximation ratio is identical to the proof of Theorem 1 by using Lemma 5, and Lemma 6 for potential function $W^{(T)}$ to obtain $\frac{F}{D} \geq (1 - 2\epsilon)$. The desired approximation ratio follows by weak duality since F and D are both feasible solutions by Proposition 3.

We now bound the number of iterations. Since we stop when there is an edge with congestion at least $\frac{1}{\epsilon^2}\ln m$. This means for all edge e, congestion is $\mathsf{cong}^{(T)}(e) = \frac{1}{c_e}\sum_p A(e,p)f_p = O(\frac{1}{\epsilon^2}\ln m)$. Whenever edge e is the bottleneck edge at time t, the new flow consumes $c(e)$

unit of e's capacity, and hence the congestion will increase by one. This is because we route $\frac{c(e)}{A(e,p^{(t)})}$ unit of flow and flow color $p^{(t)}$ consumes $A(e,p^{(t)})$ unit of e's capacity. Hence, e can be the bottleneck edge at most $O(\frac{1}{\epsilon^2}\ln m)$ time. Since there are m edges, the number of iterations is at most $O(\frac{1}{\epsilon}m\log m)$.

◀

4 Conclusion

We provide a direct stronger proof of the classic result by Garg-Könemann. The key novelty is in the analysis where we bound the primal and dual gap directly using dual objective as a potential function. We believe this our result is elementary and self-contained.

Acknowledgement:

We thank Parinya Chalermsook and Thatchaphol Saranurak for helpful discussions, encouragements and pointers to relevant literature.

References

1. Zeyuan Allen-Zhu and Lorenzo Orecchia. Nearly-linear time positive lp solver with faster convergence rate. In *Proceedings of the Forty-seventh Annual ACM Symposium on Theory of Computing*, STOC '15, pages 229–236, New York, NY, USA, 2015. ACM. doi:10.1145/2746539.2746573.
2. Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(6):121–164, 2012. doi:10.4086/toc.2012.v008a006.
3. C. Chekuri and K. Quanrud. Approximating the held-karp bound for metric tsp in nearly-linear time. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 789–800, Oct 2017. doi:10.1109/FOCS.2017.78.
4. Chandra Chekuri, T.S. Jayram, and Jan Vondrak. On multiplicative weight updates for concave and submodular function maximization. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science*, ITCS '15, pages 201–210, New York, NY, USA, 2015. ACM. doi:10.1145/2688073.2688086.
5. Chandra Chekuri and Kent Quanrud. Near-linear time approximation schemes for some implicit fractional packing problems. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '17, pages 801–820, Philadelphia, PA, USA, 2017. Society for Industrial and Applied Mathematics.
6. Chandra Chekuri and Kent Quanrud. Randomized mwu for positive lps. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '18, pages 358–377, Philadelphia, PA, USA, 2018. Society for Industrial and Applied Mathematics.
7. L. Fleischer. Approximating fractional multicommodity flow independent of the number of commodities. *SIAM Journal on Discrete Mathematics*, 13(4):505–520, 2000. doi:10.1137/S0895480199355754.
8. N. Garg and J. Könemann. Faster and simpler algorithms for multicommodity flow and other fractional packing problems. *SIAM Journal on Computing*, 37(2):630–652, 2007. doi:10.1137/S0097539704446232.
9. C. Koufogiannakis and N. E. Young. Beating simplex for fractional packing and covering linear programs. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pages 494–504, Oct 2007. doi:10.1109/FOCS.2007.62.

10 Aleksander Madry. Faster approximation schemes for fractional multicommodity flow problems via dynamic graph algorithms. In *Proceedings of the Forty-second ACM Symposium on Theory of Computing*, STOC '10, pages 121–130, New York, NY, USA, 2010. ACM. doi:10.1145/1806689.1806708.

11 S. A. Plotkin, D. B. Shmoys, and E. Tardos. Fast approximation algorithms for fractional packing and covering problems. In *[1991] Proceedings 32nd Annual Symposium of Foundations of Computer Science*, pages 495–504, Oct 1991. doi:10.1109/SFCS.1991.185411.