Lipschitz Continuous Algorithms for Graph Problems

Soh Kumabe
Graduate School of Information Science and Technology
The University of Tokyo
Tokyo, Japan
soh_kumabe@mist.i.u-tokyo.ac.jp

Yuichi Yoshida
Principles of Informatics Research Division
National Institute of Informatics
Tokyo, Japan
yyoshida@nii.ac.jp

Abstract—Graph algorithms are widely used for decision making and knowledge discovery. To ensure their effectiveness, it is essential that their output remains stable even when subjected to small perturbations to the input because frequent output changes can result in costly decisions, reduced user trust, potential security concerns, and lack of replicability. In this study, we consider the Lipschitz continuity of algorithms as a stability measure and initiate a systematic study of the Lipschitz continuity of algorithms for (weighted) graph problems.

Depending on how we embed the output solution to a metric space, we can think of several Lipschitzness notions. We mainly consider the one that is invariant under scaling of weights, and we provide Lipschitz continuous algorithms and lower bounds for the minimum spanning tree problem, the shortest path problem, and the maximum weight matching problem. In particular, our shortest path algorithm is obtained by first designing an algorithm for unweighted graphs that are robust against edge contractions and then applying it to the unweighted graph constructed from the original weighted graph.

Then, we consider another Lipschitzness notion induced by a natural mapping from the output solution to its characteristic vector. It turns out that no Lipschitz continuous algorithm exists for this Lipschitz notion, and we instead design algorithms with bounded pointwise Lipschitz constants for the minimum spanning tree problem and the maximum weight bipartite matching problem. Our algorithm for the latter problem is based on an LP relaxation with entropy regularization.

Index Terms—Approximation algorithms, Graph algorithms, Lipschitz continuity, Sensitivity

I. INTRODUCTION

Graph algorithms are widely used for decision making and knowledge discovery. To ensure their effectiveness, it is essential that their output remains stable even when subjected to small perturbations to the input. Failure to do so may result in several problems:

- The output is often used for resource allocation, and it is important to avoid frequent changes as it can result in additional costs.
- Frequent changes to decisions made by the algorithm can erode user trust.
- Inconsistency of the output in critical applications such as financial or medical can lead to unsafe decisions being made.
- The lack of consistency in the output during the process of knowledge discovery can significantly contribute to the difficulty of replicating scientific findings.

Unfortunately, many fundamental graph algorithms are vulnerable to small perturbations to the input. For example, consider Dijkstra’s algorithm, a classical algorithm for the shortest path problem that can be found in any textbook. Let \( G \) be an edge-weighted graph consisting of two vertices \( s \) and \( t \) connected by two disjoint paths \( P_1 \) and \( P_2 \) with almost the same but different lengths. Given \( s \) and \( t \) as source and target vertices, respectively, Dijkstra’s algorithm outputs the shorter path from \( s \) to \( t \), say, \( P_1 \). However, if the length of an edge in \( P_1 \) is slightly increased such that \( P_2 \) becomes shorter than \( P_1 \), then Dijkstra’s algorithm will output \( P_2 \). This indicates the vulnerability of Dijkstra’s algorithm to input perturbations, which is not desirable for some applications because the output path is often used to determine a transportation route and changing it might incur a huge cost.

In this study, we consider mitigating the effects of input perturbations on weighted graph algorithms. As a stability measure, we consider the Lipschitz continuity of graph algorithms, which guarantees that a small change in the input does not significantly change the output. Then, we design algorithms with small Lipschitz constants for various graph problems.

Before presenting our algorithmic results, we discuss how the Lipschitz continuity of a graph algorithm must be defined.
A. Lipschitz Continuity

a) Metrics: Let \( \mathcal{A} \) be an algorithm that, given a graph \( G = (V, E) \) and a weight vector \( w \in \mathbb{R}^E_{\geq 0} \) over edges, outputs an edge set \( \mathcal{A}(G, w) \subseteq E \). Then, how should we define the Lipschitz constant of \( \mathcal{A} \), or more specifically, which metric should we impose on the input and output spaces?

In this study, we always adopt the \( \ell_1 \) metric for the input space, that is, the distance between two weight vectors \( w, w' \in \mathbb{R}^E_{\geq 0} \) is defined to be \( \|w - w'\|_1 \). We do so because for combinatorial problems, it is natural to assume that the distance between \( w \) and \( w' \) is calculated as the sum of the distances between \( w(e) \) and \( w'(e) \) over \( e \in E \), and the \( \ell_1 \) metric satisfies this property.

We also use the \( \ell_1 \) metric for the output space. Depending on how we map the output edge set to a vector in the \( \ell_1 \) space, we can think of the following two variations.

(Unweighted mapping)

We map an edge set \( F \subseteq E \) to the characteristic vector \( 1_F \in \mathbb{R}^E \) of \( F \), where \( 1_F(e) = 1 \) if \( e \in F \) and 0 otherwise. Then for two edge sets \( F, F' \subseteq E \), we define
\[
d_u(F, F') := \|1_F - 1_{F'}\|_1 = |F \triangle F'|.
\]

(Wighted mapping)

We map an edge set \( F \subseteq E \) to a vector \( \sum_{e \in F} w(e)1_e \) using the weight vector \( w \in \mathbb{R}^E_{\geq 0} \), where \( 1_e \in \mathbb{R}^E \) is the characteristic vector of \( e \in E \), that is, \( 1_e(f) = 1 \) if \( f = e \) and 0 otherwise. Then for two edge sets \( F, F' \subseteq E \) and weight vectors \( w, w' \in \mathbb{R}^E_{\geq 0} \), we define
\[
d_w((F, w), (F', w')) := \left\| \sum_{e \in F} w(e)1_e - \sum_{e \in F'} w'(e)1_e \right\|_1
= \sum_{e \in F \cap F'} |w(e) - w'(e)|
+ \sum_{e \in F \setminus F'} w(e) + \sum_{e \in F' \setminus F} w'(e).
\]

We note that \( d_w((F, w), (F', w')) = \sum_{e \in F \triangle F'} w(e) \) holds.

To understand the difference between the unweighted and weighted mappings, consider the shortest path problem, where the graph and weight vector model the road network and the time required to pass through roads, respectively, and the output path represents the roads used in the trip. The distance \( d_u \) considers how many roads are changed between two trips, whereas the distance \( d_w \) considers the time spent on different roads between the two trips.

The weighted mapping is more natural than the unweighted one for Lipschitzness. To see this, let us consider a shortest path algorithm \( \mathcal{A} \). It is natural to ask \( \mathcal{A} \) to output the same path regardless of whether the distance is measured in kilometers or miles. This implies that \( \mathcal{A} \) is scale invariant, that is, it outputs the same path when edge weights are multiplied by a constant. Let \( G = (V, E) \) be a graph and \( w \in \mathbb{R}^E_{\geq 0} \) be a weight vector measured in kilometers, and consider another weight vector \( w' \in \mathbb{R}^E_{\geq 0} \) obtained from \( w \) by setting \( w'(e) = w(e) + \delta \), where \( e \in E \) and \( \delta > 0 \) is measured in kilometers. Then, if we measure the distance between outputs using the weighted mapping, the relative change of the output is
\[
d_w((\mathcal{A}(G, w), w), (\mathcal{A}(G, w'), w'))/\delta.
\]

And the relative change of the output is
\[
d_w((\mathcal{A}(G, w/c), (\mathcal{A}(G, w'/c))/c)
= \frac{d_w((\mathcal{A}(G, w), w), (\mathcal{A}(G, w'), w'))}{\delta}
= \frac{d_w((\mathcal{A}(G, w), w), (\mathcal{A}(G, w'), w'))}{\delta},
\]

and hence the two relative changes coincide. We do not have this property if we use the unweighted mapping. Hence, we first focus on the weighted mapping and then discuss the unweighted one later.

b) First Attempt: Lipschitz continuity of deterministic algorithms: Using the metric based on the weighted mapping imposed on the input and output spaces as mentioned previously, we can define the Lipschitz constant of a deterministic algorithm as follows:

Definition 1.1 (Lipschitz constant of a deterministic algorithm). Let \( \mathcal{A} \) be a deterministic algorithm that, given a graph \( G = (V, E) \) and a weight vector \( w \in \mathbb{R}^E_{\geq 0} \), outputs an edge set \( \mathcal{A}(G, w) \subseteq E \). Then, the Lipschitz constant of the algorithm \( \mathcal{A} \) on a graph \( G = (V, E) \) is
\[
\sup_{w, w' \in \mathbb{R}^E_{\geq 0}, w \neq w'} \frac{d_w((\mathcal{A}(G, w), w), (\mathcal{A}(G, w'), w'))}{\|w - w'\|_1}.
\]

Note that we only take the supremum over weight vectors and not over underlying graphs. To explain why we adopt this definition, let us consider the shortest path problem again. The weight vector can frequently change owing to traffic jams or inclement weather, whereas the
underlying graph may change because of construction or disasters, which occur less frequently. Hence, it would be more useful to consider the former type of changes than the latter.

Another reason for not taking the supremum over pairs of graphs is that the change in the underlying graph often forces any (reasonable) algorithm to change its output drastically, and hence it is impossible to bound the Lipschitz constant if we allow changes in the underlying graph. For example, consider an instance of the shortest path problem such that there are two disjoint paths—one short and the other long—between source and target vertices. Any algorithm with a reasonable approximation guarantee must output the shorter path. However, if an edge in the shorter path is removed, the algorithm must change its output to the longer path.

Unfortunately, even though we do not take the supremum over underlying graphs in Definition I.1, any (reasonable) deterministic algorithm for the shortest path problem is not Lipschitz continuous:

**Theorem I.2.** Any deterministic algorithm for the shortest path problem with a finite approximation ratio is not Lipschitz continuous, that is, its Lipschitz constant is unbounded.

To see the reason, consider a graph having two disjoint paths between the source and target vertices and the transition from a weight vector for which the first path is shorter to one for which the second path is shorter. Because the algorithm is deterministic, there is some point in the transition where the output path discontinuously changes from the first path to the second one, which implies that the algorithm is not Lipschitz.

c) **Second Attempt: Lipschitz continuity of randomized algorithms:** To remedy the aforementioned issue, we consider the Lipschitz continuity of randomized algorithms. First, we extend $d_w$, which is a metric over outputs, to a metric over output distributions. For two probability distributions $F, F'$ over subsets of $E$, the *earth mover’s distance* between $F$ and $F'$ is defined as

$$
\text{EM}_w((F, w), (F', w')) := \min_{D} \mathbb{E}_{(F, F') \sim D} d_w((F, w), (F', w')),
$$

where the minimum is taken over couplings of $F$ and $F'$, that is, distributions over pairs of sets such that its marginal distributions on the first and second coordinates are equal to $F$ and $F'$, respectively. We note that $\text{EM}_w$ coincides with $d_w$ if the distributions $F$ and $F'$ are supported by single edge sets.

For a randomized algorithm $\mathcal{A}$, a graph $G = (V, E)$, and a weight vector $w \in \mathbb{R}_{\geq 0}^E$, let $\mathcal{A}(G, w)$ denote the (random) output of $\mathcal{A}$ on $G$ and $w$. Abusing the notation, we often identify it with its distribution. Then, we define the Lipschitz constant of a randomized algorithm as follows:

**Definition I.3** (Lipschitz constant of a randomized algorithm). Let $\mathcal{A}$ be a randomized algorithm that, given a graph $G = (V, E)$ and a weight vector $w \in \mathbb{R}_{\geq 0}^E$, outputs a (random) edge set $\mathcal{A}(G, w) \subseteq E$. Then, the Lipschitz constant of the algorithm $\mathcal{A}$ on a graph $G = (V, E)$ is

$$
\sup_{w, w' \in \mathbb{R}_{\geq 0}^E, w \neq w'} \frac{\text{EM}_w((\mathcal{A}(G, w), w), (\mathcal{A}(G, w'), w'))}{\|w - w'\|_1}.
$$

We say that $\mathcal{A}$ is *Lipschitz continuous* if its Lipschitz constant is bounded for any graph $G = (V, E)$ and is $L$-Lipschitz if its Lipschitz constant is at most $L$.

If the algorithm is deterministic, then this definition coincides with Definition I.1.

Consider again the graph having two disjoint paths between the source and target vertices and the transition from a weight vector for which the first path is shorter to one for which the second path is shorter. Then, the output of a randomized algorithm can also make a transition from a distribution with most of its mass on the first path to one with most of its mass on the second path, and hence we can alleviate the issue of deterministic algorithms. However, designing Lipschitz continuous algorithms is a nontrivial task because we need to bound the ratio in Definition I.3 for any pair of weight vectors, which can be very close.

As we will discuss in Section I-B, we can design randomized Lipschitz continuous algorithms for various graph problems. However, it is not still apparent whether bounding Lipschitz constant of a randomized algorithm is practically relevant because, even if the output distributions for two weighted graphs $(G, w)$ and $(G, w')$ are close, the paths we obtain by running the algorithm on $(G, w)$ and $(G, w')$ independently may be completely different.

d) **Third Attempt: Lipschitz constant of randomized algorithms with shared randomness:** To overcome the issue mentioned above, we consider randomized algorithms such that the outputs for two similar weighted graphs are close in expectation over the internal randomness. More specifically, for a randomized algorithm $\mathcal{A}$, let $A_\pi$ denote the deterministic algorithm obtained from $\mathcal{A}$ by fixing its internal randomness to $\pi$. Then, we
define the Lipschitz constant with shared randomness of $\mathcal{A}$ as follows:

**Definition 1.4** (Lipschitz constant of a randomized algorithm with shared randomness). Let $\mathcal{A}$ be a randomized algorithm that, given a graph $G = (V, E)$ and a weight vector $w \in \mathbb{R}^E_{\geq 0}$, outputs a (random) edge set $\mathcal{A}(G, w) \subseteq E$. Then, the Lipschitz constant under shared randomness of the algorithm $\mathcal{A}$ on a graph $G = (V, E)$ is

$$\sup_{w, w' \in \mathbb{R}^E_{\geq 0}, \pi} \mathbb{E} \left[ d_w \left( (\mathcal{A}_\pi(G, w), (\mathcal{A}_\pi(G, w'), w')) \right) \right],$$

where $\pi$ represents the internal randomness of $\mathcal{A}$. We say that $\mathcal{A}$ is **Lipschitz continuous under shared randomness** if its Lipschitz constant under shared randomness is bounded for any graph $G = (V, E)$ and is L-**Lipschitz under shared randomness** if its Lipschitz constant with shared randomness is at most $L$.

Note that if a randomized algorithm is L-Lipschitz under shared randomness, then it is L-Lipschitz, and hence the former is a stronger property.

Suppose that we have a randomized algorithm $\mathcal{A}$ for the shortest path problem with a small Lipschitz constant under shared randomness and we want to compute short paths on a graph with varying weights in a stable fashion. By fixing the randomness $\pi$ first and then using $\mathcal{A}_\pi$ throughout, we can guarantee that the output paths for two similar weights are similar in expectation over $\pi$.

### B. Lipschitz Continuous Algorithms for Graph Problems

In this section, we discuss the Lipschitz continuity of randomized algorithms with shared randomness for several graph problems. Our results are summarized in Table 1.

**Problem** | Approx. Ratio | Lipschitz Constant | Reference
--- | --- | --- | ---
Min Spanning Tree | $1 + \epsilon$ | $O(\epsilon^{-1})$ | Sec. II
Shortest Path | $1 + \epsilon$ | $O(\epsilon^{-1} \log^3 n)$ | Sec. IV
Max Weight Matching | $1/8 - \epsilon$ | $O(\epsilon^{-1})$ | Sec. V

**Table I**

RESULTS FOR LIPSCHITZ CONTINUITY UNDER SHARED RANDOMNESS. $n$ represents the number of vertices in the input graph, and $\epsilon, \alpha \in (0, 1)$ are arbitrary constants.

---

- **a) Minimum spanning tree.** In the (weighted) minimum spanning tree problem, we are given a (connected) undirected graph $G = (V, E)$, and a weight vector $w \in \mathbb{R}^E_{\geq 0}$, and the goal is to output a spanning tree $T \subseteq E$ that minimizes the total weight $\sum_{e \in T} w(e)$. We show that for any $\epsilon > 0$, there exists a polynomial-time $(1 + \epsilon)$-approximation algorithm for the minimum spanning tree problem with Lipschitz constant $O(\epsilon^{-1})$ under shared randomness. To understand this upper bound, suppose that $w$ is $\{0, 1\}$-valued and that we change the value of $w(f)$ from 1 to 0 for some edge $f \in E$. This is essentially equivalent to contracting the edge $f$, and the upper bound indicates that we only need to change $O(\epsilon^{-1})$ edges in the spanning tree, which is far smaller than the spanning tree size, i.e., $\Theta(n)$. We complement the upper bound by showing that any (randomized) $(1 + \epsilon)$-approximation algorithm for the minimum spanning tree problem must have Lipschitz constant $\Omega(\epsilon^{-1})$ (even without shared randomness).

- **b) Shortest path:** In the (weighted) shortest path problem, we are given an undirected graph $G = (V, E)$, two vertices $s, t \in V$, and a weight vector $w \in \mathbb{R}^E_{\geq 0}$, and the goal is to output the shortest path between $s$ and $t$, where the length of a path $P \subseteq E$ is $\sum_{e \in P} w(e)$. We show that for any $\epsilon \in (0, 1)$, there exists a polynomial-time $(1 + \epsilon)$-approximation algorithm for the shortest path problem with Lipschitz constant $O\left(\epsilon^{-1} \log^3 n\right)$ under shared randomness, where $n$ represents the number of vertices in the input graph. Our algorithm may output a walk, i.e., the same edge may be used a multiple times in the output. We regard a walk $P$ as a multiset of edges, and we map it to a vector $\sum_{e \in P} w(e) \mathbb{1}_e$, where an edge $e \in E$ appears in the sum the same number of times that it appears in the walk $P$. Then, the distance $d_w(\cdot, \cdot)$ for walks and the Lipschitz constant of an algorithm that outputs a walk can be naturally defined. To understand the upper bound, suppose that $w$ is $\{0, 1\}$-valued and that we change the value of $w(f)$ from 1 to 0 for some edge $f \in E$. This is essentially equivalent to contracting the edge $f$, and the upper bound indicates that we only need to change $O(\epsilon^{-1} \log^3 n)$ edges in the output path, which is nontrivially small when the shortest path from $s$ to $t$ is $\omega(\log^3 n)$. We also show that any (randomized) $(1 + \epsilon)$-approximation algorithm for the shortest path problem must have Lipschitz constant $\Omega(\epsilon^{-1})$ (even without shared randomness), which implies that our upper bound is tight up to a polylogarithmic factor in $n$.

- **c) Maximum weight matching:** In the maximum weight matching problem, given a graph $G = (V, E)$ and
a weight vector \( w \in \mathbb{R}^E \geq 0 \), we want to find a matching \( M \subseteq E \) with the maximum weight, i.e., \( \sum_{e \in M} w(e) \).

We show that for any \( \epsilon > 0 \), there exists a polynomial-time \( (1/8 - \epsilon) \)-approximation algorithm with Lipschitz constant \( O(\epsilon^{-1}) \) with shared randomness. To understand this upper bound, suppose again that \( w \) is \( \{0, 1\} \)-valued and that we change the value of \( w(f) \) from 1 to 0 for some \( f \in E \). This is essentially equivalent to deleting the edge \( f \), and the upper bound indicates that we only need to change \( O(\epsilon^{-1}) \) edges in the matching, which is nontrivially small when the matching size is \( \omega(1) \). We also show that any (randomized) \( \alpha \)-approximation algorithm for the maximum weight matching problem must have Lipschitz constant \( \Omega(\alpha) \) (even without shared randomness).

We note that the proof of Theorem 1.2 can be easily extended to the minimum spanning tree problem and the maximum weight matching problem, and hence randomness is necessary to obtain Lipschitz continuous algorithms for them.

C. Pointwise Lipschitz Continuity for Unweighted Mapping

In this section, we discuss Lipschitz continuity in the case where the distances between outputs are measured using the unweighted mapping. First, we define the earth mover’s distance between output distributions \( \mathcal{F} \) and \( \mathcal{F}' \) with respect to the unweighted mapping as follows:

\[
EM_u(\mathcal{F}, \mathcal{F}') := \min_{\mathcal{D}} \mathbb{E}_{(F, F') \sim \mathcal{D}} d_u(F, F'),
\]

where the minimum is taken over couplings of \( \mathcal{F} \) and \( \mathcal{F}' \).

We cannot hope that a scale-invariant algorithm has a bounded Lipschitz constant with respect to the unweighted mapping. To see this, let \( w, w' \in \mathbb{R}^E \geq 0 \) be arbitrary weight vectors. Then for any constant \( c > 0 \), we have

\[
d_u(\mathcal{A}(G, w/c), \mathcal{A}(G, w'/c)) = \frac{c \cdot d_u(\mathcal{A}(G, w), \mathcal{A}(G, w'))}{\|w - w'\|_1},
\]

which implies that the Lipschitz constant is unbounded. Hence, we consider the following variant that look at the relative change in a local neighborhood:

**Definition 5.5** (Pointwise Lipschitz constant of a randomized algorithm with shared randomness with respect to the unweighted mapping). Let \( \mathcal{A} \) be a randomized algorithm that, given a graph \( G = (V, E) \) and a weight vector \( w \in \mathbb{R}^E \), outputs a (random) edge set \( \mathcal{A}(G, w) \subseteq E \). Then, the **pointwise Lipschitz constant under shared randomness** of the algorithm \( \mathcal{A} \) on a graph \( G = (V, E) \) at a weight vector \( w \in \mathbb{R}^E \geq 0 \) with respect to the unweighted mapping is

\[
\limsup_{w' \in \mathbb{R}^E \geq 0, w' \to w} \mathbb{E}_{\pi} \left[ \frac{d_u(\mathcal{A}_\pi(G, w), \mathcal{A}_\pi(G, w'))}{\|w - w'\|_1} \right],
\]

where \( \pi \) represents the internal randomness of \( \mathcal{A} \).

In contrast to Lipschitz constant, the pointwise Lipschitz constant can depend on the weight vector \( w \), and hence a non-scale-invariant algorithm can have a bounded pointwise Lipschitz constant.

We consider the pointwise Lipschitz continuity of algorithms with respect to the unweighted mapping for the minimum spanning tree problem and the maximum weight bipartite matching problem. Our results are summarized in Table II. Below, we discuss them in detail.

For any \( \epsilon > 0 \), we show that there exists a polynomial-time \( (1 + \epsilon) \)-approximation algorithm for the minimum spanning tree problem with pointwise Lipschitz constant \( O(\epsilon^{-1} n/\text{opt}) \) under shared randomness, where \( n \) is the number of vertices in the input graph and \( \text{opt} \) is the minimum weight of a spanning tree. As discussed previously, the dependency on \( \text{opt} \) (or some other function depending on edge weights) is unavoidable. Suppose \( w \in \mathbb{R}^E \geq 0 \) is \( \{0, 1\} \)-valued. Then, the bound shows that the change in the output tree is smaller than the tree size, \( n - 1 \), when \( \text{opt} = \omega_n(\epsilon^{-1}) \).

In the maximum weight bipartite matching problem, given a complete bipartite graph \( G = (U \cup V, E = U \times V) \) and a weight vector \( w \in \mathbb{R}^E \geq 0 \), the goal is to output a matching \( M \subseteq E \) that maximizes its weight, i.e., \( \sum_{e \in M} w(e) \). For this problem, we

| Problem                      | Approx. Ratio | Lipschitz Constant | Reference     |
|------------------------------|---------------|--------------------|---------------|
| Min Spanning Tree            | \( 1 + \epsilon \) | \( O\left(\frac{\epsilon^{-1} n}{\text{opt}}\right)\) | Sec. VII-A    |
| Max Weight Bipartite Matching| \( \frac{1}{2} - \epsilon \) | \( O\left(\frac{\epsilon^{-1} n^{3/2} \log m}{\text{opt}}\right)\) | Sec. VII-B    |

**Table II**

Results for pointwise Lipschitz continuity with shared randomness with respect to the unweighted mapping. For the minimum spanning tree problem, \( n \) represents the number of vertices in the input graph, and for the maximum weight bipartite matching problem, \( n \) and \( m \) represent the number of vertices in the left and right parts of the input bipartite graph, respectively. \( \epsilon \in (0, 1) \) is an arbitrary constant, and \( \text{opt} \) represents the optimal value.
show that there exists a polynomial-time $(1/2 - \epsilon)$-approximation algorithm with pointwise Lipschitz constant $O(\epsilon^{-1}n^{3/2} \log m / \text{opt})$ under shared randomness, where $n$ and $m$ are the numbers of vertices in the left and right parts of the input bipartite graph, respectively, and opt is the maximum weight of a matching. Suppose $n < m$ and the weight vector $w \in \mathbb{R}_{\geq 0}^E$ is $\{0,1\}$-valued. Then, the bound shows that the change in the output matching is smaller than the maximum matching size, $n$, when $\text{opt} = \omega(\epsilon^{-1/2}\sqrt{n} \log m)$.

We note that, in general, a Lipschitz continuous algorithm does not imply an algorithm with a bounded pointwise Lipschitz constant, and vice versa.

**D. Application: Online Algorithm with Small Recourse**

Let $\mathcal{P}$ be a weighted graph problem for which, given a weighted graph $(G = (V,E), w)$, the goal is to output a (feasible) edge set $F \subseteq E$ that minimizes or maximizes the objective function $f_{\mathcal{P}} : 2^E \rightarrow \mathbb{R}_{\geq 0}$. Let us consider the online version of $\mathcal{P}$ in which a graph $G = (V,E)$ is given at the beginning, and then weight vectors $w_1, \ldots, w_T \in \mathbb{R}_{\geq 0}^E$ are given online. Then, our goal is to design a (possibly randomized) online algorithm that outputs solutions $F_1, \ldots, F_T$, where $F_t$ solely depends on $G$ and $w_1, \ldots, w_t$, such that $F_t$ is a solution to the weighted graph $(G, w_t)$ and the recourse $\sum_{t=2}^T d_w((F_t, w_t), (F_{t-1}, w_{t-1}))/\|w_t - w_{t-1}\|_1$ is small.

Online algorithms with small recourse have been studied for various problems such as the set cover problem [12], the facility location problem [3], the edge coloring problem [2], and the $k$-clustering problem [17], though they studied the setting that, at each time step, a small part of the input is given instead of a weight vector.

We can use a Lipschitz continuous algorithm $\mathcal{A}$ under shared randomness to design an online algorithm with a small recourse: Fix the randomness $\pi$ first, and then run the deterministic algorithm $\mathcal{A}_\pi$ on the weighted graph $(G, w_t)$ to obtain a solution $F_t$ for every $t = 1, \ldots, T$. Then, from the Lipschitzness guarantee of $\mathcal{A}$, we can bound the expected recourse as in the following theorem:

**Theorem I.6.** Suppose that there exists a randomized $\alpha$-approximation algorithm for a weighted graph problem $\mathcal{P}$ that is $L$-Lipschitz under shared randomness. Then, there exists an $\alpha$-approximation algorithm for the online version of $\mathcal{P}$ whose expected recourse is at most $LT$.

Combining Theorem I.6 with our Lipschitz continuous algorithms, we immediately obtain online algorithms with small recourse for the minimum spanning tree problem, the shortest path problem, and the maximum weight matching problem.

**E. Related Work**

**a) Worst-case and average sensitivity.** Lipschitz continuity is closely related to the sensitivity of algorithms introduced in [20], [24]. The worst-case and average sensitivities of a randomized algorithm $\mathcal{A}$ on an (unweighted) graph $G = (V,E)$ are defined as

$$\max_{e \in E} \text{EM}_w(\mathcal{A}(G), \mathcal{A}(G \setminus e))$$

and

$$\frac{1}{|E|} \sum_{e \in E} \text{EM}_w(\mathcal{A}(G), \mathcal{A}(G \setminus e)),$$

respectively, where $G \setminus e$ is the graph obtained from $G$ by deleting the edge $e \in E$. Clearly, the average sensitivity is bounded from above by the worst-case sensitivity. The sensitivity of algorithms has been investigated for various graph problems including the minimum cut problem [24], the maximum matching problem [24], [30], and spectral clustering [21]. It is known that there is an algorithm with $o(n)$ worst-case/average sensitivity for the shortest path problem [24]. As the definition of sensitivity (1) can be easily generalized, other non-graph problems such as dynamic programming problems [15], [16] and Euclidean $k$-means [28] have also been studied from the viewpoint of sensitivity.

To see the connection to Lipschitz continuity, suppose that in the supremum of Definition I.3, we fix $w$ to be the all-one vector $1_E$ and we restrict the domain of $w'$ to $\{0,1\}$-valued vectors. Then by the triangle inequality, the (modified) Lipschitz constant on a graph $G = (V,E)$ can be bounded from above as

$$\max_{w' \in \{0,1\}^E} \frac{\text{EM}_w((\mathcal{A}(G, 1_E), 1_E), (\mathcal{A}(G, w'), w'))}{\|1_E - w'\|}$$

(2)

$$= \max_{F \subseteq E} \frac{\text{EM}_w((\mathcal{A}(G, 1_E), 1_E), (\mathcal{A}(G, 1_E \setminus F), 1_E \setminus F))}{|F|}$$

$$\leq \max_{e \in E} \text{EM}_w((\mathcal{A}(G, 1_E), 1_E), (\mathcal{A}(G, 1_E \setminus \{e\}), 1_E \setminus \{e\})).$$

(3)

For the shortest path problem, the weighted graph $(G, w)$ for a $\{0,1\}$-valued weight vector $w$ is equivalent to the graph obtained from $G$ by contracting edges $e \in E$ with $w(e) = 0$. In particular, the weighted graph $(G, 1_{E \setminus \{e\}})$ is equivalent to $G/e$, which is the graph obtained from $G$ by contracting the edge $e$. Hence, (3) can be seen as a variant of the worst-case sensitivity, where the operation applied to the graph is edge contraction instead of edge deletion, that is,

$$\max_{e \in E} \text{EM}_w(\mathcal{A}(G), \mathcal{A}(G/e)).$$

(4)
Indeed, our Lipschitz continuous algorithm for the shortest path problem is based on an algorithm with a bounded sensitivity with respect to edge contraction.

A notable difference between the sensitivity and the Lipschitz constant is that the former is trivially bounded by the maximum solution size whereas it is not clear a priori whether there is an algorithm for which the latter is bounded.

b) Replicability: Note that, when an algorithm has a small Lipschitz constant under shared randomness, it has the ability to replicate the output with respect to weight perturbations. Another form of replicability has been introduced for the stochastic setting by Impagliazzo et al. [14]. Consider two sets of $n$ i.i.d. elements, $X = (x_1, ..., x_n)$ and $Y = (y_1, ..., y_n)$. Then, a randomized algorithm $A$ is called $\rho$-replicable for $\rho > 0$ if $\Pr_{X,Y,\pi}[A_X(X) = A_Y(Y)] \geq 1 - \rho$, where $\pi$ denotes the internal randomness of $A$. Impagliazzo et al. [14] showed a replicable algorithm for learning halfspaces, and since then, replicable algorithms have been developed for bandit problems [7] and clustering problems [8].

c) Adversarial attacks and Lipschitz continuity of neural networks: In the field of machine learning, it is known that the predictions of the trained model can be significantly changed by slight perturbations of the input, and such perturbations are called adversarial attacks [10], [22]. Because adversarial attacks threaten the security of machine learning-based systems, there is active research on training models that are robust against them [4], [18]. See surveys [1], [27], [31] for more details.

To mitigate the effects of adversarial attacks, neural networks with small Lipschitz constants have been proposed and their properties have been investigated [5], [9], [23], [25], [26]. It is also reported that bounding Lipschitz constants of neural networks stabilizes the training process and often produces models with better output quality [11], [19], [29].

We note that bounding the Lipschitz constant of a neural network is often easy because it is bounded by the product of the Lipschitz constants of the activation functions and linear transformations used in the neural network, which are easy to calculate. However, to design Lipschitz continuous algorithms for graph problems, we need to bound approximation ratio and Lipschitz constant simultaneously, and we often need nontrivial techniques as we will see in this paper.

F. Technical Overview

We first discuss designing Lipschitz continuous algorithms (without shared randomness) and then briefly discuss the necessary modifications that ensure Lipschitz continuity under shared randomness.

a) Minimum spanning tree: It is known that Kruskal’s algorithm has (worst-case) sensitivity $O(1)$ against edge deletions [24]. However, it is not Lipschitz continuous because it is deterministic (see Theorem I.2), and hence some modification is required.

Our Lipschitz continuous algorithm for the minimum spanning tree problem works as follows: Given a graph $G = (V, E)$ and a weight vector $w \in \mathbb{R}^E_0$, we sample $\hat{w}(e)$ uniformly from $[w(e), (1+\epsilon)w(e)]$ for each edge $e \in E$, and then apply Kruskal’s algorithm to the new weight vector $\hat{w}$. This algorithm clearly achieves $(1+\epsilon)$-approximation.

We can show that, to bound the Lipschitz constant, it suffices to consider a pair of weight vectors $(w, w')$ such that $w'$ is obtained from $w$ by setting $w'(e) = w(e) + \delta$ for some $e \in E$ and $\delta > 0$. Then, the total variation distance between $\hat{w}$ and $\hat{w}'$ is $O(\epsilon^{-1}\delta/w(e))$, where $\hat{w}'$ is constructed from $\hat{w}$ in the same way as $\hat{w}$ is constructed from $w$. Then, we can define a coupling, i.e., a joint distribution, between $\hat{w}$ and $\hat{w}'$ such that $\hat{w} \neq \hat{w}'$ with probability $O(\epsilon^{-1}\delta/w(e))$. Also, we can show that when $w \neq w'$ occurs in the coupling, the distance between the output spanning trees is $O(w(e))$. This implies that the earth mover’s distance is $O(\epsilon^{-1}\delta)$ and hence the Lipschitz constant is $O(\epsilon^{-1})$.

Our algorithm and analysis for pointwise Lipschitz-ness with respect to the unweighted mapping is similar though we need some care because we use the optimal value to determine the range from which we sample $\hat{w}(e)$ and it varies depending on the weight vector.

b) Shortest path: To obtain Lipschitz continuous algorithm for the shortest path problem, we first design an algorithm for the (unweighted) shortest path problem with a low sensitivity with respect to edge contraction (see (4)). We will use this algorithm as a subroutine in our Lipschitz continuous algorithm.

The subroutine takes an unweighted graph $\hat{G}$ and two vertices $s, t \in V(\hat{G})$ as the input and returns an approximate $s$-$t$ shortest path in $\hat{G}$. This subroutine is recursive: It samples a vertex $v$ called a pivot, recursively computes $s$-$v$ and $v$-$t$ walks that are nearly optimal, and then returns the walk obtained by concatenating the two walks. We choose the pivot $v$ so that it is roughly in the middle of a nearly optimal $s$-$t$ path. By doing so, we
can bound the depth of the recursion by $O(\log n)$ and guarantee that the output walk is nearly optimal.

Now, we turn to analyzing the sensitivity of the subroutine with respect to edge contraction. Let $e \in E(\hat{G})$ and we want to bound the earth mover’s distance between the output distributions for $\hat{G}$ and $\hat{G}/e$. Informally, we say that a recursion call for computing a $u$-$v$ shortest path is active if there is a nearly optimal $u$-$v$ path passing through $e$ and is inactive otherwise. Then, we can show the following:

- recursion calls invoked in an inactive recursion call are also inactive, and
- with high probability, at most one of the two recursion calls invoked in an active recursion call is active.

These properties imply that the expected number of active recursion calls in each recursion depth is $O(1)$. Also, we can prove that in an inactive call, the pivot is sampled from exactly the same distributions for $\hat{G}$ and $\hat{G}/e$, and thus it does not contribute to the sensitivity at all. Additionally, we can show that each active call contributes to the sensitivity by $O(\log^2 n)$. Regarding that the number of active calls is $O(\log n)$, the sensitivity of the algorithm can be bounded by $O(\log^3 n)$.

To obtain a Lipschitz continuous algorithm for the shortest path problem, we construct an unweighted graph $\hat{G}(w)$ from the input weighted graph $(G, w)$, and apply the subroutine to it to obtain a walk in $\hat{G}(w)$, and then output the corresponding walk in $G$. Here, the graph $\hat{G}(w)$ is obtained by replacing each edge of the input graph $G$ with a path of suitable length so that each path in $G$ naturally corresponds to a path in $\hat{G}(w)$. The length of each path is sampled from a certain distribution so that the total variation distance between $\hat{G}(w)$ and $\hat{G}(w')$ is proportional to $\|w - w'\|_1$. Then, we can bound the Lipschitz continuity of the algorithm by using the sensitivity of the subroutine.

c) Maximum weight matching: Our algorithm is based on the algorithm for the maximum weight matching proposed by Yoshida and Zhou [30]. For a parameter $\alpha > 2$, their algorithm first classifies edges $e$ according to the value $[\log_{\alpha} w(e)]$. Then, it runs a randomized greedy to compute a matching for each edge class, and then returns a matching obtained by combining them.

Although their algorithm has a bounded weighted sensitivity, a discrete analogue of Lipschitz constant, its Lipschitz constant is not bounded. This is because an arbitrarily small change in the weight of an edge may cause the edge to be classified into a different class. To resolve this issue, we sample a parameter $b \in [1, \alpha]$ and classify edges $e$ according to the value $[\log_{\alpha} w(e)]$. Then, the total variation distance between the classifications obtained from weight vectors $w$ and $w'$ is proportional to $\|w - w'\|_1$, and we can bound the Lipschitz constant.

d) Maximum weight bipartite matching: In contrast to the previous Lipschitz continuous algorithm for the maximum weight matching problem, our pointwise Lipschitz continuous algorithm for the maximum weight bipartite matching problem is based on linear programming (LP). The standard LP relaxation for the maximum weight bipartite matching problem is not stable against perturbations to the edge weight. Hence, we consider LP with entropy regularization [6]. Although entropy regularization was originally introduced to speed up the computation of the earth mover’s distance, we use it here to stabilize the computation of LP. For a weight vector $w \in \mathbb{R}_{\geq 0}^E$, let $LP_{ent}(w)$ denote the LP for the weighted graph $(G, w)$ with entropy regularization. Then, we can show that (i) the solution to $LP_{ent}$ is nearly optimal to the original LP, and (ii) for any weight vector $w'$, the $\ell_1$ distance between the solutions of $LP_{ent}(w)$ and $LP_{ent}(w')$ is proportional to $\|w - w'\|_1$. Then, we carefully round the obtained fractional solution to an integral one in such a way that for any two fractional solutions $x$ and $x'$, the earth mover’s distance between the (random) integer solutions obtained from $x$ and $x'$ is proportional to $\|x - x'\|_1$ with respect to the unweighted mapping.

e) Lipschitz continuity under shared randomness: In our algorithm, all randomness is caused by sampling processes from continuous or discrete distributions. For example, in our Lipschitz continuous algorithm for the minimum spanning tree, randomness is caused by sampling each edge’s modified weight $\tilde{w}(e)$ from a uniform distribution over $[w(e), (1 + \epsilon)w(e)]$. Let us consider fixing this randomness using shared randomness $\pi$.

Let $F$ be a function that maps weight vectors $w \in \mathbb{R}_{\geq 0}^E$ to probability distributions, and suppose that in our algorithm, we sample values from the distribution $F(w)$ at some point. We define a sampling process $SAMPLE(\cdot, \pi)$ to be stable for $F$ if it satisfies the following two conditions:

- For any $w \in \mathbb{R}_{\geq 0}^E$, when $\pi$ is sampled from a uniform distribution, the output distribution of $SAMPLE(F(w), \pi)$ is equal to $F(w)$.
- For any $w, w' \in \mathbb{R}_{\geq 0}^E$, when $\pi$ is sampled from a uniform distribution, the probability that $SAMPLE(F(w), \pi)$ and $SAMPLE(F(w'), \pi)$ differ is proportional to the total variation distance between
\( \mathcal{F}(w) \) and \( \mathcal{F}(w') \).

By replacing all sampling processes in the algorithm with stable processes, we can make our algorithm Lipschitz continuous under shared randomness.

There are four types of sampling processes in the algorithm. Three of them sample from a continuous uniform distribution \( \mathcal{F}(w) = U([l(w), r(w)]) \), where \( U[a, b] \) denotes the uniform distribution over \([a, b] \), with \( l(w) \) and \( r(w) \) being constant functions, with \( l(w) \) and \( r(w) \) having a constant ratio, and with \( l(w) \) and \( r(w) \) having a constant difference. The last one samples from a discrete distribution with fixed support.

For each of these processes, we construct a stable algorithm. Furthermore, by discussing how to modify each part of the five algorithms mentioned above, we obtain Lipschitz continuous algorithms under shared randomness with the same approximation ratio and Lipschitz constant, as well as polynomial-time complexity.

G. Preliminaries

For a function \( f : X \to Y \) and \( S \subseteq X \), let \( f|_S : S \to Y \) denote the function obtained from \( f \) by restricting its domain to \( S \).

We use bold symbols to denote random variables. For a random variable \( X \) and an event \( E \), we use \( (X | E) \) to denote \( X \) conditioned on \( E \). We often identify a random variable with its distribution.

For two random variables \( X \) and \( X' \), the total variation distance between them is given as follows:

\[
\text{TV}(X, X') := \min_{D} \Pr_{(x, x') \sim D}[x = x'],
\]

where the minimum is taken over couplings between \( X \) and \( X' \), that is, distributions over pairs such that its marginal distributions on the first and the second coordinates are equal to \( X \) and \( X' \), respectively.

For an element \( e \in E \), we use \( 1_f \in \mathbb{R}^E \) to denote the characteristic vector of \( f \), that is, \( 1_f(e) = 1 \) and \( 1_f(e) = 0 \) for \( e \in E \setminus \{f\} \). The following lemma indicates that, to bound the Lipschitz constant, it suffices to consider pairs of weight vectors that differ by one coordinate. The proof is deferred to Appendix A.

**Lemma I.7.** Suppose that there exist some \( c > 0 \) and \( L > 0 \) such that

\[
\text{EM}_w((A(G, w), w), (A(G, w + \delta 1_f), w + \delta 1_f)) \leq \delta L
\]

holds for all \( w \in \mathbb{R}_{\geq 0}^E, f \in E \) and \( 0 < \delta \leq c \). Then, \( A \) is \( L \)-Lipschitz.

---

**Algorithm 1:** Lipschitz-continuous algorithm for the minimum spanning tree problem

| Procedure LipMST \( (G = (V, E), w, \epsilon) \) |
| --- |
| for \( e \in E \) do |
| Sample \( \hat{w}(e) \) uniformly from \([w(e), (1 + \epsilon)w(e)]\); |
| return arbitrary minimum spanning tree of \( G \) with respect to \( \hat{w} \). |

---

II. Minimum Spanning Tree

In this section, we consider the Lipschitz continuity of the minimum spanning tree problem and prove the following:

**Theorem II.1.** For any \( \epsilon > 0 \), there exists a polynomial-time \((1 + \epsilon)\)-approximation algorithm for the minimum spanning tree problem with Lipschitz constant \( O(\epsilon^{-1}) \).

Our algorithm is simple. Given a graph \( G = (V, E) \), a weight vector \( w \in \mathbb{R}_{\geq 0}^E \) and \( \epsilon > 0 \), we sample \( \hat{w}(e) \) uniformly from \([w(e), (1 + \epsilon)w(e)]\) for each \( e \in E \), and then return an arbitrary minimum spanning tree with respect to the weight vector \( \hat{w} \) (see LipMST in Algorithm 1).

First, we analyze the approximation ratio.

**Lemma II.2.** The approximation ratio of LipMST is (at most) \( 1 + \epsilon \).

**Proof.** Let \( T^* \) be the optimal solution with respect to the weight vector \( w \), and let \( T \) be the output of LipMST. Let \( \text{opt} \) (resp., \( \hat{\text{opt}} \)) be the minimum weight of a spanning tree with respect to \( w \) (resp., \( \hat{w} \)). Then, we have

\[
\hat{\text{opt}} = \sum_{e \in T} \hat{w}(e) \leq \sum_{e \in T^*} \hat{w}(e) \leq \sum_{e \in T^*} (1 + \epsilon)w(e) = (1 + \epsilon)\text{opt},
\]

where the first inequality is from the fact that \( T \) is a minimum spanning tree with respect to \( \hat{w} \), and the claim holds.

Now we analyze the Lipschitz continuity of LipMST. For a weight vector \( w \in \mathbb{R}_{\geq 0}^E \) with distinct elements, the minimum spanning tree with respect to \( w \) uniquely exists, and we denote it by \( T(w) \). The following lemma is simple but useful.

**Lemma II.3.** Let \( f \in E \) be an edge, \( \delta > 0 \), and \( w \in \mathbb{R}_{\geq 0}^E \) be such that both \( w \) and \( w + \delta 1_f \) have distinct elements. Then, \( T^{w + \delta 1_f} := T(w + \delta 1_f) \) is either identical
to $T := T(w)$ or is obtained from $T$ by deleting $f$ and adding an edge with weight at most $w(f) + \delta$.

**Proof.** If deleting $f$ disconnects $G$, we have $T^δ_f = T$. Suppose otherwise. Let $T^−_f$ be the minimum spanning tree of $G^−_f := (V, E \setminus \{f\})$ with respect to the weight vector $w|_{E \setminus \{f\}}$. Then, $T$ (resp., $T^δ_f$) is obtained from $T^−_f \cup \{f\}$ by deleting the (unique) edge $g$ (resp., $g^δ_f$) of the largest weight with respect to the weight vector $w$ (resp., $w + \delta 1_f$) in the unique cycle $C$ in $T^−_f \cup \{f\}$.

If $g = f$, then the weight of $f$ is the largest in $C$ with respect to $w$, and hence it is the largest with respect to $w + \delta 1_f$ too. Therefore, we have $T = T^δ_f = T^−_f$. If $g^δ_f \neq f$, then the weight of $f$ is not the largest in $C$ with respect to $w + \delta 1_f$, and hence it is not the largest with respect to $w$ either. Therefore, the edges with the largest weight in $C$ with respect to $w$ and $w + \delta 1_f$ are the same, and hence we have $T = T^δ_f$. If $g \neq f$ and $g^δ_f = f$, then $T$ uses $f$, and $T^δ_f$ is obtained from $T$ by deleting $f$ and adding an edge with the largest weight in $C$ with respect to $w$, which is at most the weight of $f$ on $w + \delta 1_f$, that is, $w(f) + \delta$.

Let $f \in E$ be an edge and $\delta > 0$. To bound the Lipschitz constant, by Lemma I.7, it suffices to bound

$$EM_w ((\text{LipMST}(G, w, \epsilon), w), \text{LipMST}(G, w + \delta 1_f, w + \delta 1_f)).$$

To bound (5), we consider a coupling (i.e., a joint distribution) $\mathcal{W}$ between $\hat{w}$ and $\hat{w}^δ_f$ defined as follows: For each weight vector $\hat{w} \in \mathbb{R}_{\geq 0}^E$, we transport the probability mass for $\hat{w} = \hat{w}$ to that for $\hat{w}^δ_f = \hat{w}$ as far as possible in such a way that the probability mass for $\hat{w}|_{E \setminus \{f\}} = \hat{w}|_{E \setminus \{f\}}$ is always transported to that for $\hat{w}^δ_f|_{E \setminus \{f\}} = \hat{w}|_{E \setminus \{f\}}$, which is possible because the distributions of $\hat{w}(e)$ and $\hat{w}^δ_f(e)$ are identical for $e \in E \setminus \{f\}$. The remaining probability mass is transported arbitrarily.

**Lemma II.4.** We have

$$Pr_{(\hat{w}, \hat{w}^δ_f) \sim \mathcal{W}} \left[ \hat{w} \neq \hat{w}^δ_f \right] = Pr_{(\hat{w}, \hat{w}^δ_f) \sim \mathcal{W}} \left[ \hat{w}(f) \neq \hat{w}^δ_f(f) \right] \leq \frac{(1 + \epsilon)\delta}{\epsilon(w(f) + \delta)}.$$

**Proof.** The equality holds by the design of the coupling $\mathcal{W}$.

Now, we show the inequality. Recall that $\hat{w}(f)$ is sampled uniformly from $[w(f), (1 + \epsilon)w(f)]$ and $\hat{w}^δ_f(f)$ is sampled uniformly from $[w(f) + \delta, (1 + \epsilon)(w(f) + \delta)]$. Hence, we have

$$Pr_{(\hat{w}, \hat{w}^δ_f) \sim \mathcal{W}} \left[ \hat{w}(f) \neq \hat{w}^δ_f(f) \right] \leq \frac{(1 + \epsilon)(w(f) + \delta) - (1 + \epsilon)w(f)}{(1 + \epsilon)(w(f) + \delta) - (w(f) + \delta)} = \frac{(1 + \epsilon)\delta}{\epsilon(w(f) + \delta)},$$

as desired.

**Lemma II.5.** LipMST is $O(\epsilon^{-1})$-Lipschitz.

**Proof.** Note that $\hat{w}$ has distinct elements with probability one and hence we only need to consider the case that it happens when calculating the Lipschitz constant. By Lemmas II.3 and II.4, we have

$$EM_w ((\text{LipMST}(G, w, \epsilon), w), \text{LipMST}(G, w + \delta 1_f, w + \delta 1_f)) \leq \frac{\mathbf{E}_w \left[ d_w \left( (T(\hat{w}), (T(\hat{w})^δ_f), w + \delta 1_f) \right) \right]}{(1 + \epsilon)(w(f) + \delta) - (1 + \epsilon)w(f)} \frac{(1 + \epsilon)\delta}{\epsilon(w(f) + \delta)} \leq \frac{2(1 + \epsilon)2\delta^2}{\epsilon} \leq O \left( \frac{\delta}{\epsilon} \right).$$

Therefore, the Lipschitz constant is bounded as

$$\sup_{\delta > 0, f \in E} EM_w ((\text{LipMST}(G, w, \epsilon), w), \text{LipMST}(G, w + \delta 1_f, w + \delta 1_f)) \leq O \left( \frac{1}{\epsilon} \right).$$

**Theorem II.1** follows by combining Lemmas II.2 and II.5.

**III. CONTRACTION SENSITIVITY OF SHORTEST PATH**

In this section, we present an algorithm for the shortest path problem that is stable against edge contractions. We will use this algorithm as a subroutine in our Lipschitz continuous algorithm for the shortest path problem in Section IV. In this section, we only consider unweighted graphs.

To define the sensitivity to edge contractions, we first formally define edge contraction. Let $G = (V, E)$ be a graph and $e = \{u, v\} \in E$ be an edge. Then, the *contraction* of an edge $e$ in $G$ is an operation that yields a graph $G/e = (V', E')$ with

$$V' = (V \setminus e) \cup \{v_e\},$$
$$E' = E \setminus \{(\{u, x\} : x \in N(u)) \cup (\{v, x\} : x \in N(v))\} \cup \{\{v_e, x\} : x \in N(u) \cup N(v) \setminus \{u, v\}\}$$

Theorem II.1 follows by combining Lemmas II.2 and II.5.
where \( v_e \) is a newly introduced vertex and \( N(a) \) denotes the set of neighbors of a vertex \( a \in V \). Then, we define the sensitivity to edge contraction for the shortest path problem as follows:

**Definition III.1.** Let \( \mathcal{A} \) be a (randomized) algorithm for the shortest path problem that, given a graph \( G = (V, E) \) and two vertices \( s, t \in V \), outputs an \( s-t \) walk. Then, the contraction sensitivity of \( \mathcal{A} \) on a graph \( G = (V, E) \) is

\[
\max_{e \in E, e \cap \{s,t\} = \emptyset} EM_u(\mathcal{A}(G), \mathcal{A}(G/e)).
\]

Recall that when calculating the distance \( EM_u \), a walk \( P = (e_1, \ldots, e_i) \) is mapped to a vector \( \sum_{i=1}^{t} 1_{e_i} \), and hence an edge that appears multiple times in \( P \) contributes by the same number of times in the vector. We only consider contracting edges not incident to \( s \) and \( t \), because we do not need to contract such edges in our Lipschitz continuous algorithm in Section IV, and this simplifies the analysis.

The goal of this section is to show the following:

**Theorem III.2.** For any \( \epsilon > 0 \), there is a \((1 + \epsilon)\)-approximation algorithm for the shortest path problem with contraction sensitivity \( O(\epsilon^{-1} \log^3 |V|) \).

Varma and Yoshida [24] showed that any algorithm with a finite approximation ratio must have \( \Omega(n) \) sensitivity with respect to edge deletion. This is because in the edge-deletion case, an edge in the output path may disappear and we may need to completely change the output path. However, in the edge-contraction case, we can use (essentially) the same path even if an edge in the path is contracted.

In Section III-A, we present our algorithm and an overview of the analysis. We analyze the approximation ratio in Section III-B and the contraction sensitivity in Sections III-C and III-D.

### A. Proof Overview

In this section, we present our algorithm for the unweighted shortest path problem (pseudocode is presented in Algorithm 2).

To construct a walk between two vertices \( s, t \in V \) in the input graph \( G = (V, E) \), we first sample a vertex \( v \in V \), which we call a pivot, such that there is an \( s-t \) path passing through \( v \) whose length is close to that of the shortest one and the sampling process of \( v \) is stable against edge contractions. Then, we recursively construct \( s-v \) and \( v-t \) walks and output an \( s-t \) walk obtained by combining them.

Before discussing the approximation and contraction sensitivity of the algorithm, we introduce useful notation.

**Algorithm 2:** Shortest path algorithm stable against edge contractions

1. Procedure **Rec**\((G, s, t, \gamma)\)
2. Sample \( d \) uniformly from \([\frac{1}{4} + 2\gamma, \frac{3}{4} - 2\gamma] \);
3. Sample \( l \) uniformly from \([\gamma, 2\gamma] \);
4. If \( \text{opt}(s, t) \leq \gamma^{-1} \) then
5. Return the \( s-t \) shortest path on \( G \) computed by breadth-first search.
6. Let \( V_{d,l} \) be the set of vertices \( v \in V \) with \( \text{opt}(s, v) \leq (d + l) \text{opt}(s, t) \) and \( \text{opt}(v, t) \leq (1 - d + l) \text{opt}(s, t) \);
7. Sample \( v \in V_{d,l} \) uniformly at random;
8. Return the walk obtained by concatenating \( \text{Rec}(G, s, v, \gamma) \) and \( \text{Rec}(G, v, t, \gamma) \).
9. Procedure **SP**\((G = (V, E), s, t, \epsilon)\)
10. Sample \( \gamma^{-1} \) uniformly from \([720\epsilon^{-1} \log |V|, 1440\epsilon^{-1} \log |V|] \);
11. Return \( \text{Rec}(G, s, t, \gamma) \).

Let \( G = (V, E) \) be an unweighted graph. For two vertices \( s, t \in V \), let \( \text{opt}_G(s, t) \) denote the length of the \( s-t \) shortest path of \( G \). For a vertex \( v \in V \), let \( \text{opt}_G(s, t, v) \) denote the length of the shortest walk from \( s \) to \( t \) passing through \( v \), and for an edge \( e \in E \), let \( \text{opt}_G(s, t, e) \) denote the length of the shortest walk from \( s \) to \( t \) passing through \( e \in E \). We call a path (resp., walk) from a vertex \( s \in V \) to one of the endpoints of an edge \( e \in E \) an \( s-e \) path (resp., \( s-e \) walk). Let \( \text{opt}_G(s, e) \) denote the length of the shortest \( s-e \) path. If the underlying graph \( G \) is clear from the context, we omit the subscripts from these symbols.

**a) Approximation Ratio:** Let us discuss the approximation ratio of the algorithm above. First, the choice of the range from which we sample \( l \) in Line 3 ensures that restricting the feasible domain to the family of \( s-t \) walks passing through the pivot \( v \) increases the optimal value only by a \((1 + O(\gamma))\) factor. Specifically, we have

\[
\text{opt}(s, t, v) = \text{opt}(s, v) + \text{opt}(v, t) \leq (1 + 2\gamma)\text{opt}(s, t) \leq (1 + 4\gamma)\text{opt}(s, t).
\]

According to the choice of the range from which we sample \( d \) in Line 2, the optimal values for the two subproblems induced by the pairs \((s,v)\) and \((v,t)\) are smaller than the original one by a constant factor. Specifically, we have

\[
\max\{\text{opt}(s, v), \text{opt}(v, t)\} \leq \frac{3}{4}\text{opt}(s, t).
\]
This implies that the recursion depth is $O(\log |V|)$. It follows that the approximation ratio is bounded by
\[
(1 + O(\gamma))^{O(\log |V|)} \leq 1 + O(\gamma \log |V|) = 1 + O(\epsilon).
\]

b) Contraction Sensitivity: Now, we consider the contraction sensitivity of the algorithm. First, we focus on a particular call of REC and show that the pivot distributions for REC($G, s, t, \gamma$) and REC($G/e, s, t, \gamma$) are close, or more specifically, their total variation distance is $O\left(\frac{\log |V|}{\gamma \cdot \text{opt}(s, t)}\right)$. To bound the earth mover’s distance in the definition of contraction sensitivity, we consider transporting the probability mass of REC($G, s, t, \gamma$) corresponding to a particular choice of the pivot $v$ to that of REC($G/e, s, t, \gamma$) corresponding to the same pivot as far as possible. We transport the remaining probability mass arbitrarily, which increases the contraction sensitivity of the algorithm by
\[
O\left(\frac{\log |V|}{\gamma \cdot \text{opt}(s, t)}\right) \cdot (1 + O(\gamma \log |V|))\cdot \text{opt}(s, t)
\]

\[
\leq O\left(\frac{\log |V|}{\gamma}\right),
\]
where $(1 + O(\gamma \log |V|))\cdot \text{opt}(s, t)$ is the upper bound on the length of the output walk.

Next, we consider the contraction sensitivity of the whole algorithm. A recursion call REC($G, s, t, \gamma$) is active if $\text{opt}_{G}(s, t, e) \leq (1 + k\gamma)\text{opt}_{G}(s, t)$ and inactive otherwise, where $k = 16$. We can show that, if REC($G, s, t, \gamma$) is inactive, then
- the distribution of the pivots chosen in REC($G, s, t, \gamma$) and REC($G/e, s, t, \gamma$) are exactly the same, and
- the recursion calls REC($G, s, \bar{v}, \gamma$) and REC($G, \bar{v}, t, \gamma$) are also inactive.

Thus, inactive recursion calls are irrelevant to the contraction sensitivity.

Assume that REC($G, s, t, \gamma$) is active and the same pivot $v$ is chosen in REC($G, s, t, \gamma$) and REC($G/e, s, t, \gamma$). Then, we can show that, with probability $1 - O(\gamma)$, at least one of the two recursion calls REC($G/e, s, v, \gamma$) and REC($G/e, v, t, \gamma$) is inactive. Intuitively, this holds because both recursion calls are active only when $|\text{opt}_{G}(s, e) - d| \leq O(\gamma)$, which occurs with probability $O(\gamma)$ because we sampled $d$ uniformly from $[\frac{1}{4} + 2\gamma, \frac{3}{4} - 2\gamma]$. Then, the number of recursion calls relevant to contraction sensitivity is bounded by $(1 + O(\gamma))^{O(\log |V|)} = O(\log |V|)$. The contraction sensitivity of the algorithm is bounded as
\[
O\left(\frac{\log |V|}{\gamma}\right) \cdot O(\log |V|) = O\left(\frac{\log^2 |V|}{\gamma}\right).
\]

B. Approximation Ratio

In this section, we analyze the approximation ratio of Algorithm 2. First, we analyze the approximation ratio of REC.

Lemma III.3. For the pivot $v$ sampled in REC($G, s, t, \gamma$), we have
\[
\text{opt}(s, v) + \text{opt}(v, t) \leq (1 + 4\gamma)\text{opt}(s, t).
\]

Proof. We have
\[
\text{opt}(s, v) + \text{opt}(v, t) \\ \leq (d + l)\text{opt}(s, t) + (1 - d + l)\text{opt}(s, t) \\ = (1 + 2l)\text{opt}(s, t) \leq (1 + 4\gamma)\text{opt}(s, t).
\]

Lemma III.4. For $\gamma > 0$, we have
\[
|\text{REC}(G, s, t, \gamma)| \leq \text{opt}(s, t)^{1+14\gamma}.
\]

Proof. We prove by induction on $\text{opt}(s, t)$. The statement clearly holds when $\text{opt}(s, t) \leq \gamma^{-1}$.

Suppose that $\text{opt}(s, t) > \gamma^{-1}$. Then, we have
\[
|\text{REC}(G, s, t, \gamma)| \\ \leq \max_{d, l, v \in V_{d, t}} (|\text{REC}(G, s, v, \gamma)| + |\text{REC}(G, v, t, \gamma)|) \\ \leq \max_{d, l, v \in V_{d, t}} (\text{opt}(s, v)^{1+14\gamma} + \text{opt}(v, t)^{1+14\gamma}) \\ = \max_{d, l, v \in V_{d, t}} (\text{opt}(s, v) \cdot \text{opt}(v, t)^{1+14\gamma} + \text{opt}(v, t) \cdot \text{opt}(v, t)^{1+14\gamma}) \\ \leq \left(\frac{3}{4}\text{opt}(s, t)^{1+14\gamma}\right) \cdot \max_{d, l, v \in V_{d, t}} (\text{opt}(s, v) + \text{opt}(v, t)) \\ \leq \left(\frac{3}{4} \cdot \text{opt}(s, t)^{1+14\gamma}\right) (1 + 4\gamma)\text{opt}(s, t) \\ = \left(\frac{3}{4} \cdot \text{opt}(s, t)^{1+14\gamma}\right) \text{opt}(s, t)^{1+14\gamma} \\ \leq \text{opt}(s, t)^{1+14\gamma},
\]
where the second inequality is from the induction hypothesis, the third inequality is from how we select the pivot, the fourth inequality is from Lemma III.3, and the last inequality is from $1 + 4\gamma \leq \left(\frac{3}{4}\right)^{14\gamma}$.

Now, we bound the approximation ratio of SP.

Lemma III.5. The approximation ratio of SP($G, s, t, \epsilon$) is (at least) $1 + \epsilon$.

Proof. By Lemma III.4, SP outputs a walk of length $\text{opt}(s, t)^{1+14\gamma}$, which is bounded as
\[
\text{opt}(s, t)^{1+14\gamma} \leq \text{opt}(s, t)^{\frac{14\gamma}{1 + \log_{10}14\gamma}} \cdot \text{opt}(s, t)
\]
\[ \leq |V| \frac{14e^{1/20}}{2|V|} \cdot \opt(s, t) \]
\[ = \exp \left( \frac{14e}{720} \right) \cdot \opt(s, t) \leq (1 + \epsilon) \opt(s, t), \]
where the first inequality is from \( \gamma \leq \frac{e}{720 \log |V|} \), the second inequality is from \( \opt(s, t) \leq |V| \), and the last inequality is from \( \exp(x) \leq 1 + 2x \), which holds for \( x \leq \frac{1}{2} \).

\[ \square \]

C. Total Variation Distance Between Pivots

In this section, we bound the total variation distance between pivots selected in a particular call of REC. Let \( v \) and \( v^e \) be the pivots sampled in REC(\( G, s, t, \gamma \)) and REC(\( G/e, s, t, \gamma \)), respectively. We denote the variables \( V, d, l \) used in REC(\( G/e, s, t, \gamma \)) as \( V^e, d^e, l^e \) to distinguish them from those used in REC(\( G, s, t, \gamma \)). Our goal is to prove the following:

**Lemma III.6.** We have
\[ TV(v, v^e) \leq \frac{4 \log n}{\gamma \cdot \opt_G(s, t)}. \]

We start with several lemmas.

**Lemma III.7.** For any \( d \) and \( l \) (in the ranges specified in REC), we have \( V^e_{d,l} \subseteq V_{d,l}^e \cup \{v_e\} \) (recall that \( v_e \) is the vertex introduced when constructing \( G/e \) from \( G \)).

**Proof.** Let \( v \in V^e_{d,l} \setminus \{v_e\} \). Then, we have
\[ \opt_G(s, v) \leq \opt_{G/e}(s, v) + 1 \]
\[ \leq (d + l) \opt_{G/e}(s, t) + 1 \]
\[ \leq (d + l) \opt_G(s, t) + 1 \]
\[ = \left( d + l + \frac{1}{\opt_G(s, t)} \right) \opt_G(s, t), \]
\[ \opt_G(v, t) \leq \opt_{G/e}(v, t) + 1 \]
\[ \leq (1 - d + l) \opt_G(s, t) + 1 \]
\[ \leq (1 - d + l) \opt_{G/e}(s, t) + 1 \]
\[ = \left( 1 - d + l + \frac{1}{\opt_G(s, t)} \right) \opt_G(s, t), \]
and the claim holds. \( \square \)

**Lemma III.8.** For any \( d \) and \( l \) (in the ranges specified in REC), we have \( |V^e_{d,l}| \geq \gamma \cdot \opt_G(s, t) \).

**Proof.** Let \( P \) be an arbitrary \( s-t \) shortest path. From the definition of \( V^e_{d,l} \), any vertex \( v' \) on \( P \) with \( (d - l) \opt_G(s, v') \leq \opt_G(s, v) \leq (d + l) \opt_G(s, t) \) is contained in \( V^e_{d,l} \). Therefore, we have
\[ |V^e_{d,l}| \geq [(d + l) \opt_G(s, t)] - [(d - l) \opt_G(s, t)] \]
\[ \geq 2 \opt_G(s, t) - 1 \geq \gamma \cdot \opt_G(s, t). \]

**Corollary III.9.** For any \( v \in V, d, l \) (in the ranges specified in REC), we have
\[ \Pr[v = v | d = d, l = l] \leq \frac{1}{\gamma \cdot \opt_G(s, t)}. \]

The next technical lemma is useful in our analysis.

**Lemma III.10.** Let \( a_0, \ldots, a_k \) be a non-decreasing array with \( a_0 > 0 \). Then,
\[ \sum_{d=0}^{k-1} \left( 1 - \frac{a_d}{a_{d+1}} \right) \leq a_k/a_0. \]

**Proof.** We prove by induction on \( k \). When \( k = 0 \), the statement clearly holds.

Suppose \( k \geq 1 \) and the statement holds for \( k - 1 \). Then, we have
\[ \sum_{d=0}^{k-1} \left( 1 - \frac{a_d}{a_{d+1}} \right) \leq \log \frac{a_{k-1}}{a_k} + \log \frac{a_k}{a_{k-1}} = \log \frac{a_k}{a_0} \]
where the last inequality is from \( 1 - \frac{1}{x} \leq \log x \) for \( x \geq 1 \).

The following lemma proves that \( TV(v, v^e) \) is small when \( d = d^e \).

**Lemma III.11.** For any \( d \) (in the range specified in REC), we have
\[ TV((v | d = d), (v^e | d^e = d)) \leq \frac{2}{\gamma \cdot \opt_G(s, t)}. \]

**Proof.** We have
\[ TV((v | d = d), (v^e | d^e = d)) \]
\[ \leq \frac{1}{\gamma} \int_{2 \gamma} TV((v | d = d, l = l), (v^e | d^e = d, l^e = l)) dl \]
\[ = \frac{1}{\gamma} \int_{2 \gamma} \left( \sum_{v \in V^e_{d,l}} \max(0, \Pr[v = v | d = d, l = l]) 
- \Pr[v = v | d^e = d, l^e = l] \right) dl \]
\[ \leq \frac{1}{\gamma} \int_{2 \gamma} \left( \Pr[v = v | d = d, l = l] 
+ \sum_{v \in V^e_{d,l}} \max(0, \left( \frac{1}{|V^e_{d,l}|} - \frac{1}{|V^e_{d,l}|} \right)) \right) dl 
\leq \frac{1}{\gamma} \int_{2 \gamma} \left( 2 \cdot \frac{1}{\gamma \cdot \opt_G(s, t)} + \sum_{v \in V^e_{d,l}} \max(0, \frac{1}{|V^e_{d,l}|} - \frac{1}{|V^e_{d,l}|}) \right) dl 
\leq \frac{2}{\gamma \cdot \opt_G(s, t)}. \]
\[
\leq \frac{2}{\gamma \cdot \text{opt}_G(s, t)} + \frac{1}{\gamma} \int_\gamma^{2\gamma} \left(1 - \frac{|V_{d,l}|}{|V_{d,l}| + \text{opt}_G(s, t)}\right) dl \\
\leq \frac{2}{\gamma \cdot \text{opt}_G(s, t)} + \gamma^{-1} \int_\gamma^{2\gamma} \left(1 - \frac{|V_{d,l}|}{|V_{d,l}| + \text{opt}_G(s, t) + \{v_c\}}\right) dl \\
+ \gamma^{-1} \int_\gamma^{2\gamma} \left(1 - \frac{|V_{d,l}|}{|V_{d,l}| + \text{opt}_G(s, t) + \{v_c\}}\right) dl,
\]
where the second inequality is from Line 6 of Algorithm 2, the third inequality is from Corollary III.9, the fourth inequality is from |V_{d,l} - e| \leq |V_{d,l}|, and the last inequality is from Lemma III.7.

Let \(c = \frac{1}{\text{opt}_G(s, t)}\). Then, we have

\[
\int_\gamma^{2\gamma} \left(1 - \frac{|V_{d,l}|}{|V_{d,l}| + \text{opt}_G(s, t) + \{v_c\}}\right) dl \\
\leq \int_\gamma^{2\gamma} \left(1 - \frac{|V_{d,l}|}{|V_{d,l}| + \text{opt}_G(s, t) + \{v_c\}}\right) dl \\
\leq \int_\gamma^{2\gamma} \left(1 - \frac{|V_{d,a+kc}|}{|V_{d,a+(k+1)c}| + 1}\right) da \\
\leq \int_\gamma^{2\gamma} \left(1 - \frac{|V_{d,a+kc}|}{|V_{d,a+(k+1)c}|}\right) da \\
+ \sum_{k=0}^{2\gamma-a} \left(1 - \frac{|V_{d,a+kc}|}{|V_{d,a+(k+1)c}|}\right) da \\
\leq \int_\gamma^{2\gamma} \left(1 + \sum_{k=0}^{2\gamma-a} \left(1 - \frac{|V_{d,a+kc}|}{|V_{d,a+(k+1)c}|}\right) da \\
= \int_\gamma^{2\gamma} \left(1 + \sum_{k=0}^{2\gamma-a} \left(1 - \frac{|V_{d,a+kc}|}{|V_{d,a+(k+1)c}|}\right) da \\
\leq \int_\gamma^{2\gamma} \left(1 + \sum_{k=0}^{2\gamma-a} \left(1 - \frac{|V_{d,a+kc}|}{|V_{d,a+(k+1)c}|}\right) da \\
\leq \int_\gamma^{2\gamma} \left(1 + \log \left(\frac{|V_{d,a+(k+1)c}|}{|V_{d,a}|}\right)\right) da \\
\leq \int_\gamma^{2\gamma} \left(1 + \log |V|\right) da = \frac{1 + \log |V|}{\text{opt}_G(s, t)},
\]
where the first inequality is from |V_{d,a+kc}| \leq |V_{d,a+(k+1)c}| and

\[
1 - \frac{x}{x+y+z} = 1 - \frac{x}{x+y} + \frac{xz}{(x+y)(x+y+z)}
\]
holds for \(x, y, z \geq 0\), the second inequality is from \(|V_{d,a+(k+1)c}| \geq \gamma \cdot \text{opt}_G(s, t)\), the third inequality is from the definition of \(c\), the fourth inequality is from Lemma III.10, and the last inequality is from \(1 \leq |V_{d,l}| \leq |V|\) for all \(l\). Therefore, we have

\[
\text{TV}((v \mid d = d), (v^e \mid d^e = d)) \\
\leq \frac{2}{\gamma \cdot \text{opt}_G(s, t)} + \frac{1}{\gamma} \cdot \frac{1 + \log |V|}{\text{opt}_G(s, t)} \\
\leq \frac{1 + \log |V|}{\gamma \cdot \text{opt}_G(s, t)}. \]

Proof of Lemma III.6. Because the distributions of \(d\) and \(d^e\) are identical, we can omit the condition \(d = d^e\) of Lemma III.11 by considering the natural probability transportation.

D. Contraction Sensitivity

In this section, we analyze the contraction sensitivity of Algorithm 2. Our goal is to prove the following.

**Lemma III.12.** The contraction sensitivity of \(\text{SP}(G, s, t, \epsilon)\) is \(O(\epsilon^{-1} \log^3 |V|)\).

Let \(e \in E\) be the edge that will be contracted. To evaluate the earth mover’s distance between \(\text{SP}(G, s, t, \epsilon)\) and \(\text{SP}(G/e, s, t, \epsilon)\), we consider transporting the probability mass of \(\text{SP}(G, s, t, \epsilon)\) corresponding to a particular choice of \(\gamma\) to that of \(\text{SP}(G/e, s, t, \epsilon)\) corresponding to the same choice of \(\gamma\). The remaining mass is transported arbitrarily. Let us fix \(\gamma\) and analyze

\[
\text{EM}_u(\text{REC}(G, s, t, \gamma), \text{REC}(G/e, s, t, \gamma)).
\]

We transport the probability mass of \(\text{REC}(G, s, t, \gamma)\) corresponding to a particular choice of the pivot \(v\) to that of \(\text{REC}(G/e, s, t, \gamma)\) corresponding to the same choice of the pivot as far as possible. For a fixed \(v\), we transport the probability mass from \(\text{REC}(G, s, v, \gamma)\) (resp., \(\text{REC}(G, v, t, \gamma)\)) to \(\text{REC}(G/e, s, v, \gamma)\) (resp., \(\text{REC}(G/e, v, t, \gamma)\)) as far as possible, where the mass is transported recursively, as in the case of \(\text{REC}(G, s, t, \gamma)\). Because \(\text{REC}(G, s, v, \gamma)\) and \(\text{REC}(G, v, t, \gamma)\) (resp., \(\text{REC}(G/e, s, v, \gamma)\) and \(\text{REC}(G/e, v, t, \gamma)\)) are independent for fixed \(v\), we obtain a probability transportation from the probability mass of \(\text{REC}(G, s, t, \gamma)\) to that of \(\text{REC}(G/e, s, t, \gamma)\) as their direct product. The remaining probability mass is transported arbitrarily.

In Section III-D1, we introduce the notion of active and inactive recursion calls, which are useful for bounding the earth mover’s distance. We prove Lemma III.12 in Section III-D2.
1) Active and Inactive Recursion Calls: Let $k = 16$. Recall that we say that a call $\text{REC}(G,s,t,\gamma)$ is active if $\text{opt}_G(s,t,e) \leq (1 + k\gamma)\text{opt}_G(s,t)$ and inactive otherwise. In this section, we present several properties of active and inactive calls of $\text{REC}$. We start with the following.

**Lemma III.13.** If $\text{REC}(G,s,t,\gamma)$ is inactive, then both $\text{REC}(G,s,v,\gamma)$ and $\text{REC}(G,v,t,\gamma)$ are inactive.

**Proof.** By symmetry, it suffices to prove the claim only for $\text{REC}(G,s,v,\gamma)$. Assume that $\text{REC}(G,s,v,\gamma)$ is active. Then,

$$\text{opt}_G(s,t,e) \leq \text{opt}_G(s,v,e) + \text{opt}_G(v,t) \leq (1 + k\gamma)\text{opt}_G(s,v) + \text{opt}_G(v,t) = k\gamma \cdot \text{opt}_G(s,v) + (\text{opt}_G(s,v) + \text{opt}_G(v,t)) \leq \frac{3}{4}k\gamma \cdot \text{opt}_G(s,t) + (1 + 4\gamma)\text{opt}_G(s,t) \leq (1 + k\gamma)\text{opt}_G(s,t),$$

where the first inequality is from the triangle inequality, the second inequality is from the assumption that $\text{REC}(G,s,v,\gamma)$ is active, the third inequality if from Line 6 of Algorithm 2, and the last inequality is from $k \geq 16$. It follows that $\text{REC}(G,s,t,\gamma)$ is active, which is a contradiction. \hfill $\square$

The next lemma shows that inactive recursive calls have no contribution to the sensitivity.

**Lemma III.14.** If $\text{REC}(G,s,t,\gamma)$ is inactive, then we have $TV(v,v^e) = 0$.

**Proof.** Because $\text{REC}(G,s,t,\gamma)$ is inactive, we have $\text{opt}_G(s,t) = \text{opt}_{G/e}(s,t)$. Therefore, we have $V_{d,t} \subseteq V_{d,t} \cup e$. For $v \in e$, we have

$$\text{opt}_{G/e}(s,v_e) + \text{opt}_{G/e}(v_t,e) \geq \text{opt}_G(s,t,e) - 2 \geq (1 + k\gamma)\text{opt}_G(s,t,e) - 2 \geq (1 + 4\gamma)\text{opt}_G(s,t) \geq 1,$$

where the last inequality is from $\gamma \cdot \text{opt}_G(s,t) \geq 1$. Therefore, $v \not\in V_{d,t}$. For the vertex $v_e$ in $G/e$ corresponding to $e$, we have

$$\text{opt}_{G/e}(s,v_e) + \text{opt}_{G/e}(e,v_t) \geq \text{opt}_G(s,e) + \text{opt}_G(e,t) \geq \text{opt}_G(s,t,e) - 2 \geq (1 + k\gamma)\text{opt}_G(s,t,e) - 2 \geq (1 + 4\gamma)\text{opt}_G(s,t) \geq 1.$$
where the first equality is from the range from which we sample \( d \), the third inequality is from the triangle inequality, the fourth inequality is from \( k\gamma \leq \frac{1}{2} \) (because \( k = 16 \) and \( \gamma \leq \frac{360}{\epsilon} \log |V| \leq 0.01 \)), and the last inequality is from the range from which we sample \( l \).

2) Proof of Lemma III.12: First we bound the sensitivity caused by sampling \( \gamma \).

**Lemma III.16.** We have

\[
\text{TV}(\gamma, \gamma^c) \leq \frac{2}{|V|}.
\]

**Proof.** We have

\[
\text{TV}(\gamma, \gamma^c) = \text{Pr}[\gamma^{-1} \in [1440\epsilon^{-1} \log(|V| - 1), 1440\epsilon^{-1} \log |V|]] = 1440\epsilon^{-1}(\log(|V|) - \log(|V| - 1)) = \frac{720\epsilon^{-1} \log |V|}{2} = 2 \left(1 - \frac{\log(|V| - 1)}{\log(|V|)}\right) \leq \frac{2}{|V|}.
\]

**Lemma III.17.** The recursion depth is at most \( 4 \log |V| \).

**Proof.** At each call of \( \text{REC} \), we have \( \max\{\text{opt}_G(s, v), \text{opt}_G(u, t)\} \leq \frac{3}{4} \text{opt}_G(s, t) \). Hence, the recursion depth is at most \( \log_{4/3} \text{opt}(s, t) \leq 4 \log \text{opt}(s, t) \leq 4 \log |V| \).

Using the lemmas above and Lemma III.6, we prove the following:

**Lemma III.18.**

\[
\mathbb{E}_u(\text{REC}(G, s, t, \gamma), \text{REC}(G/e, s, t, \gamma)) \leq \frac{1}{72} |V|^{14\gamma} (|V|^{360 \gamma} - 1) \log |V|.
\]

**Proof.** First, we bound the expected number of active recursive calls. We have

\[
\mathbb{E}[\text{The number of active recursive calls}] \leq \sum_{k=0}^{\frac{4 \log |V|}{\epsilon}} (1 + 72\gamma)^k = \frac{\left((1 + 72\gamma)^{4 \log |V|} + 1\right)}{72\gamma} \leq \frac{\left((1 + 72\gamma)^{5 \log |V|} - 1\right)}{72\gamma} = \frac{\left(|V|^{5 \log (1 + 72\gamma)} - 1\right)}{72\gamma} \leq \frac{|V|^{360 \gamma} - 1}{72\gamma}.
\]

where the first inequality is from Lemmas III.15 and III.17, the first equality is from

\[
\sum_{k=0}^{a} (1 + x)^k = \left((1 + x)^a - (1 + x)^{-1}\right) \cdot \frac{1}{1 - 1+x}.
\]

and the last inequality is from \( \log(1 + x) \leq x \).

From Lemmas III.4 and III.6, each active recursion call \( \text{REC}(G, s', t', \gamma) \) contributes to \( \text{EM}_u(\text{REC}(G, s, t, \gamma), \text{REC}(G/e, s, t, \gamma)) \) by

\[
\text{opt}_G(s', t')^{1 + 14\gamma} \cdot \frac{4 \log |V|}{\gamma \cdot \text{opt}_G(s', t')}.
\]

Thus, we have

\[
\text{EM}_u(\text{REC}(G, s, t, \gamma), \text{REC}(G/e, s, t, \gamma)) \leq \mathbb{E} \left[ \sum_{\text{REC}(G, s', t', \gamma) \text{ is active}} \text{opt}_G(s', t')^{1 + 14\gamma} \cdot \frac{4 \log |V|}{\gamma \cdot \text{opt}_G(s', t')} \right]
\]

\[
\leq \frac{|V|^{360 \gamma} - 1}{72\gamma} \cdot |V|^{14\gamma} \cdot \frac{4 \log |V|}{\gamma}
\]

\[
= \frac{4}{72\gamma^2} |V|^{14\gamma} (|V|^{360 \gamma} - 1) \log |V|
\]

\[
\leq \frac{1}{72\gamma^2} |V|^{14\gamma} (|V|^{360 \gamma} - 1) \log |V|,
\]

where the second inequality is from \( \text{opt}_G(s', t') \leq |V| \).

**Proof of Lemma III.12.** If \( \epsilon^{-1} > \text{opt}_G(s, t) \), then the claim clearly holds because the sensitivity is at most \( \text{opt}_G(s, t) = O(\epsilon^{-1} \log^3 |V|) \). Therefore, we assume that \( \epsilon^{-1} \leq \text{opt}_G(s, t) \).

\[
\mathbb{E}_u(\text{SP}(G, s, t, \epsilon), \text{SP}(G/e, s, t, \epsilon)) \leq \frac{1}{\gamma^2} |V|^{14\gamma} (|V|^{360 \gamma} - 1) \log |V| + \frac{2}{|V|} \cdot (1 + \epsilon) \text{opt}(s, t)
\]

\[
\leq \frac{1}{\gamma^2} |V|^{14\gamma} (|V|^{360 \gamma} - 1) \log |V| + 4
\]

\[
\leq 4147200 \log^3 |V| + 4 = O \left( \frac{\log^3 |V|}{\epsilon} \right),
\]
where the first inequality is from Lemmas III.16 and III.18, and the last inequality is from
\[
\frac{1}{\gamma^2}|V|^{14+\gamma} \left( |V|^{360\gamma} - 1 \right) \log |V| \\
\leq \left( \frac{1440 \log |V|}{\epsilon} \right)^2 \cdot |V| \left( \frac{14e}{720 \log |V|} \right) \\
\cdot \left( |V|^{360\gamma} - 1 \right) \cdot \log |V| \\
= \left( \frac{1440 \log |V|}{\epsilon} \right)^2 \exp \left( \frac{14e}{720} \right) \\
\cdot \left( \exp \left( \frac{360e}{720} \right) - 1 \right) \cdot \log |V| \\
\leq \left( \frac{1440 \log |V|}{\epsilon} \right)^2 \cdot (1 + \epsilon) \cdot \epsilon \cdot \log |V| \\
\leq \frac{41472000 \log^3 |V|}{\epsilon}.
\]
Here, the first inequality is from $\frac{1440 \log |V|}{\epsilon} \leq \gamma \leq 1 + 2x$, which holds for $x \leq \frac{1}{2}$, and the last inequality is from $\epsilon \leq 1$. \hfill \Box

**Proof of Theorem III.2.** The proof follows from Lemmas III.5 and III.12. \hfill \Box

**IV. Lipschitz Continuity of Shortest Path**

In this section, we provide our algorithm for the shortest path problem with a bounded Lipschitz constant and prove the following:

**Theorem IV.1.** For any $\epsilon \in (0, 1)$, there exists a polynomial-time $(1 + \epsilon)$-approximation algorithm for the shortest path problem with Lipschitz constant $O(\epsilon^{-1} \log^3 n)$, where $n$ is the number of vertices in the input graph.

Our idea is to convert the input weighted graph $(G, w)$ into an unweighted graph $\hat{G}$ by replacing each edge with a path whose length is determined by its weight, apply Algorithm 2 on $\hat{G}$, and then convert the obtained walk on $\hat{G}$ back to a walk on $G$.

However, it is unclear how to convert a walk in $\hat{G}$ back to a walk in $G$ in such a way that a small contraction sensitivity of the former implies a small Lipschitz constant of the latter because the former may walk around in a path in $\hat{G}$ corresponding to an edge in $G$. To resolve this issue, we replace each edge of $\hat{G}$ with two directed paths oriented in opposite directions. Then, we can naturally map a walk in $\hat{G}$ to a walk in $G$ because a walk entering one end of a path must leave from the other end. For this idea to work, we extend Algorithm 2 for directed graphs. We discuss this directed version in detail in Section IV-A.

We describe our algorithm in Section IV-B, and analyze its approximation and Lipschitz continuity in Sections IV-C and IV-D, respectively.

**A. Directed Shortest Path**

In the (unweighted) directed shortest path problem, we are given an unweighted directed graph $G = (V, E)$, a source vertex $s \in V$, and a target vertex $t \in V$, we want to compute a directed path from $s$ to $t$ with the shortest length. We extend the algorithm in Section III for the directed shortest path problem.

We must be careful when discussing contraction sensitivity for the directed shortest path problem because the contraction of a directed edge may affect the reachability relation between vertices and significantly change the optimal values. To avoid this complex issue that is unnecessary for our reduction, we call an edge $(u, v) \in E$ in a directed graph $G = (V, E)$ contractible if $d^+(u) = 1$ and $d^+(v) = 1$, where $d^+(a)$ and $d^-(a)$ denote the outdegree and indegree, respectively, of a vertex $a \in V$, and only consider contraction of contractible edges. Note that contracting contractible edges do not change reachability relations of vertices (except for the deleted and newly introduced vertices). Then, we define the sensitivity for directed graphs to edge contraction as follows:

**Definition IV.2.** Let $A$ be a (randomized) algorithm that, given a directed graph $G = (V, E)$, outputs an edge (multi)set $F \subseteq E$. Then, the directed contraction sensitivity of $A$ on a directed graph $G = (V, E)$ is

\[
\max_{e \in E: \text{contractible}} \text{EM}_{e}(A(G), A(G/e)).
\]

Now, we can show the following:

**Lemma IV.3.** For any $\epsilon > 0$, there exists a $(1 + \epsilon)$-approximation algorithm for the directed shortest path problem with directed contraction sensitivity $O(\epsilon^{-1} \log^3 |V|)$.

The algorithm of Lemma IV.3 is obtained from Algorithm 2 by replacing $\text{opt}_{G}(u, v)$ with the directed counterpart. The analysis of the approximation ratio and contraction sensitivity is almost identical to that for Algorithm 2. We use contractibility of edges only to prove Lemmas III.7 and III.14. In the proof of Lemma III.7, we can use contractibility to show that $\text{opt}_{G}(s, v) \leq \text{opt}_{G/e}(s, v) + 1$ and $\text{opt}_{G}(v, t) \leq \text{opt}_{G/e}(v, t) + 1$. Next, we discuss required changes to the proof of Lemma III.14. For a (directed) edge $e$, let $\text{opt}_{G}(s, e)$
Algorithm 3: Lipschitz continuous algorithm for the shortest path problem

1. **Procedure** LipSP\( (G = (V,E), w, s,t,\epsilon) \)
2. \( \text{if } \text{opt}_{G,w}(s,t) = 0 \text{ then} \)
3. \( \text{return an arbitrary } s-t \text{ path with length } 0. \)
4. Sample \( b \) uniformly from \( \left[ \frac{w(e) + \text{opt}_{G,w}(s,t)}{e|V|}, \frac{w(e) + \text{opt}_{G,w}(s,t)}{e|V|} \right] \).
5. Let \( V = V, E = \emptyset \).
6. for \( e = (u,v) \in E \) do
7. \( \text{Let } l(e) = \left\lfloor \frac{w(e)}{b} \right\rfloor; \)
8. Sample \( x(e) \) uniformly from \([0,1)\); \( \text{Let } \hat{w}(e) = l(e) + 2 \) if \( x(e) \leq \frac{(l(e)+1)b - w(e)}{b} \) and \( \hat{w}(e) = l(e) + 3 \) otherwise;
9. \( \text{if } \hat{w}(e) \leq 12e^{-1}|V| + 3 \text{ then} \)
10. \( \hat{V} \leftarrow \hat{V} \cup \{e_1, \ldots, e_{\hat{w}(e)-1} \} \cup \{e_{\hat{w}(e)-1} \}; \)
11. \( \hat{E} \leftarrow \hat{E} \cup \{(u = e_0, e_1), \ldots, (e_{\hat{w}(e)-1}, e_{\hat{w}(e)} = v) \} \cup \{v = e_{\hat{w}(e)-1}, e_{\hat{w}(e)} \}; \)
12. return the corresponding path in \( G \) of \( \text{DiSP}(\hat{G} = (\hat{V}, \hat{E}), s,t,\epsilon/4) \).

\( \hat{b} \) (resp., \( \text{opt}_G(e,t) \)) denote the length of the shortest path from \( s \) to the tail of \( e \) (resp., from the head of \( e \) to \( t \)).

In the calculation of (7), we can replace the first inequality with \( \text{opt}_G(s,v) + \text{opt}_G(v,t) \geq \text{opt}_G(s,t) \) for \( v \in e \), which holds because contractibility ensures that a path that passes through \( v \) also passes through \( e \). Moreover, in the calculation of (9), we can replace the first equality with \( \text{opt}_G(s,e) + \text{opt}_G(e,t) = \text{opt}_G(s,t) + 1 \). To see this, note that \( \text{opt}_{G/e}(s,v_e) = \text{opt}_{G}(s,e) \) holds because contractibility ensures that any path from \( s \) to the head of \( e \) passes through \( e \), and similarly, \( \text{opt}_{G/e}(v_e,t) = \text{opt}_{G}(e,t) \) holds, implying the equality.

**B. Algorithm Description**

Our algorithm, LipSP, is presented in Algorithm 3, where DiSP is the shortest path algorithm for directed graphs presented in Section IV-A. This algorithm first transforms the given weighted graph \( (G = (V,E), w) \) into a (random and unweighted) directed graph \( \hat{G} \) and then applies DiSP to it. To construct \( \hat{G} \), we first sample a discretization parameter \( b \) and then round the length of each edge \( e \in E \) to an integer \( \hat{w}(e) \) that is roughly \( l(e) := \left\lfloor \frac{w(e)}{b} \right\rfloor \) (we discuss the exact choice later). Then, \( \hat{G} \) is constructed by replacing each edge \( e = \{u,v\} \) in \( G \) with two directed paths of lengths \( \hat{w}(e) \), one is from \( u \) to \( v \), and the other is from \( v \) to \( u \).

We construct a directed graph instead of an undirected graph because we want to transform the output of DiSP back to an \( s-t \) walk on \( G \). Because DiSP outputs a walk (not necessarily a path), if \( G \) is undirected, then the corresponding “walk” in \( G \) may turn around in the middle of an edge.

To attain Lipschitz continuity, we need some more tricks. First, we ignore lengthy edges in \( G \) (Line 10). By doing this, we can bound the number of vertices in \( \hat{G} \), and keep the contraction sensitivity of \( \text{DiSP}(\hat{G}, s,t,\epsilon/4) \) low. This also makes the algorithm run in strongly polynomial time. Second, we choose \( \hat{w}(e) \) from \( l(e) + 2 \) or \( l(e) + 3 \), not \( l(e) \) or \( l(e) + 1 \). Thus, we can ensure that reducing the weight of an edge in \( G \) always corresponds to contracting a contractible edge in \( \hat{G} \).

**C. Approximation Ratio**

We analyze the approximation ratio of Algorithm 3. We denote the minimum length of a directed \( s-t \) path in \( (G,w) \) as \( \text{opt}_{G,w}(s,t) \).

**Lemma IV.4.** The approximation ratio of LipSP is \( 1 + \epsilon \).

**Proof.** Let \( P \) be a path that attains \( \text{opt}_{G,w}(s,t) \). Note that \( P \) does not include any edge \( e \) with \( \hat{w}(e) > 12e^{-1}|V| + 3 \), because we have \( w(e) \geq (\hat{w}(e) - 3)b > 12e^{-1}|V|b \geq \text{opt}_{G,w}(s,t) \). Therefore, we can take a path \( P' \) in \( \hat{G} \) corresponding to \( P \). Then, we have

\[
\text{LipSP}(G, w, s,t,\epsilon) \\
\leq \left( 1 + \frac{\epsilon}{4} \right) \text{opt}_G(s,t) b \\
\leq \left( 1 + \frac{\epsilon}{4} \right) |E(P')| b \\
\leq \left( 1 + \frac{\epsilon}{4} \right) \sum_{e \in E(P)} \left( \left\lfloor \frac{w(e)}{b} \right\rfloor + 3 \right) b \\
\leq \left( 1 + \frac{\epsilon}{4} \right) \left( \sum_{e \in E(P)} w(e) + 3b |E(P)| \right) \\
= \left( 1 + \frac{\epsilon}{4} \right) \left( \text{opt}_{G,w}(s,t) + 3b |E(P)| \right) \\
\leq \left( 1 + \frac{\epsilon}{4} \right) \left( \text{opt}_{G,w}(s,t) + \frac{|E(P)| \text{opt}_{G,w}(s,t)}{2|V|} \right) \\
\leq \left( 1 + \frac{\epsilon}{4} \right) \left( 1 + \frac{\epsilon}{2} \right) \text{opt}_{G,w}(s,t),
\]
where the first inequality is from Lemma III.5 and the fact that \(w(e) \leq \tilde{w}(e)b\) holds for all \(e \in E\), the second inequality is from the fact that each edge \(e\) in \(G\) corresponds to a path in \(\tilde{G}\) with length at most \(\left[\frac{w(e)}{b}\right] + 3\), and the last inequality is from \(\epsilon \leq 1\). □

D. Lipschitz Continuity

Next, we analyze the Lipschitz continuity of Algorithm 3. Let \(\delta > 0\) and \(f \in E\). By Lemma 1.7, it suffices to bound

\[
\frac{1}{\delta} \text{EM}_{w}(\text{LIPSP}(G, w, s, t, \epsilon), \text{LIPSP}(G, w + \delta 1_f, s, t, \epsilon))
\]

for sufficiently small \(\delta > 0\). Specifically, we assume \(\delta \leq \frac{\epsilon \cdot \text{opt}_{G,w}(s, t)}{12|V|}\). To distinguish the variables used in \(\text{LIPSP}(G, w, s, t, \epsilon)\) from those used in \(\text{LIPSP}(G, w + \delta 1_f, s, t, \epsilon)\), we add superscripts \(\delta f\) to the latter, e.g., \(b^{\delta f}\).

We consider the coupling \(\mathcal{D}\) between \((b, x)\) and \((b^{\delta f}, x^{\delta f})\) defined as follows: For each parameter \(b \in \mathbb{R}_\geq\), we transport the probability mass for \(b = b\) to that for \(b^{\delta f} = b\) as far as possible in such a way that for each choice of \(x \in [0, 1]\), the probability mass for \(x = x\) is transported to that for

\[
x^{\delta f}(e) = \begin{cases} x(e) & \text{if } e \in E \setminus \{f\}, \\ x(e) - \frac{\delta}{b} & \text{if } e = f, x(f) > \frac{\delta}{b}, \\ x(e) - \frac{\delta}{b} + 1 & \text{if } e = f, x(f) \leq \frac{\delta}{b}. \end{cases} \tag{11}
\]

The remaining mass is transported arbitrarily. Clearly, (11) is a one-to-one correspondence from \([0, 1]\) to itself for all \(e \in E\). Therefore, any pair \(((b, x), (b^{\delta f}, x^{\delta f}))\) in the support of the remaining mass satisfies \(b \neq b^{\delta f}\).

We begin by presenting the following useful property of the coupling.

Lemma IV.5. Let \(((b, x), (b^{\delta f}, x^{\delta f}))\) be a pair sampled from \(\mathcal{D}\) conditioned on \(b = b^{\delta f}\). Then we have \(\tilde{w}(e) = \tilde{w}^{\delta f}(e) - 1\) if \(e = f\) and \(x(e) \leq \frac{\delta}{b}\) hold, and we have \(\tilde{w}(e) = \tilde{w}^{\delta f}(e)\) otherwise.

We defer the proof to Section IV-D1 and continue the analysis of the Lipschitz continuity of LIPSP.

When \(\tilde{w}(e) = \tilde{w}^{\delta f}(e)\) holds, \(G\) and \(G^{\delta f}\) are exactly the same and so are \(\text{LIPSP}(G, w, s, t, \epsilon)\) and \(\text{LIPSP}(G, w + \delta 1_f, s, t, \epsilon)\). The case of \(\tilde{w}(e) = \tilde{w}^{\delta f}(e) - 1\) can be divided into the following three cases:

(i) \(b \neq b^{\delta f}\);
(ii) \(b = b^{\delta f}\), and the two paths corresponding to \(f\) exist in \(G\) but do not exist in \(G^{\delta f}\); and
(iii) \(b = b^{\delta f}\), and the two paths corresponding to \(f\) exist in both \(G\) and \(G^{\delta f}\), but the length of the paths in \(G\) is smaller than that in \(G^{\delta f}\) by 1.

Let \(p_\neq\), \(p_{\text{path}}\), and \(p_{\text{hen}}\) be the probabilities that Cases (i), (ii), and (iii) occur, respectively, in the coupling \(\mathcal{D}\). We will bound these probabilities. First, we consider Case (i).

Lemma IV.6. We have

\[
p_\neq \leq \frac{2\delta}{\text{opt}_{G,w}(s, t)}.
\]

Proof. We have

\[
p_\neq = TV(b, b^{\delta f})
\leq \Pr \left[ b^{\delta f} \in \left[ \frac{\epsilon \cdot \text{opt}_{G,w}(s, t)}{6|V|}, \frac{\epsilon \cdot \text{opt}_{G,w+\delta 1_f}(s, t)}{6|V|} \right] \right]
\leq \frac{\delta}{6|V|} \cdot \frac{12|V|}{\text{opt}_{G,w+\delta 1_f}(s, t)} = \frac{2\delta}{\text{opt}_{G,w+\delta 1_f}(s, t)},
\]

where the first inequality is from \(\delta > 0\) and hence the interval of \(b\) is to the left of that of \(b^{\delta f}\), and the second inequality is from \(\text{opt}_{G,w+\delta 1_f}(s, t) \leq \text{opt}_{G,w}(s, t) + \delta\). □

Next we consider Case (ii).

Lemma IV.7. We have

\[
p_{\text{path}} \leq \frac{6\delta}{\text{opt}_{G,w}(s, t)}.
\]

Proof. Assume \(b = b^{\delta f}\). Case (ii) happens only when \(\tilde{w}(f) = \lceil 12\epsilon^{-1}|V| \rceil + 3\) and \(\tilde{w}^{\delta f}(f) = \lceil 12\epsilon^{-1}|V| \rceil + 4\). This implies \(l(f)\) is either \(\lceil 12\epsilon^{-1}|V| \rceil\) or \(\lceil 12\epsilon^{-1}|V| \rceil + 1\), and it follows that

\[
\frac{w(f)}{b} - 1 \leq \frac{w(f)}{b} = l(f) \leq \frac{12|V|}{\epsilon} + 1 \leq \frac{12|V|}{\epsilon} + 1,
\]

\[
\frac{w(f)}{b} \geq \frac{w(f)}{b} = l(f) \geq \frac{12|V|}{\epsilon} \geq \frac{12|V|}{\epsilon} - 1.
\]

Equivalently, we have

\[
b \geq \left[ \frac{w(f)}{12\epsilon^{-1}|V| + 2 \cdot \frac{w(f)}{12\epsilon^{-1}|V| - 1} } \right].
\]

Therefore, we have

\[
p_{\text{path}} \leq \Pr \left[ b = b^{\delta f} \in \left[ \frac{w(f)}{12\epsilon^{-1}|V| + 2 \cdot \frac{w(f)}{12\epsilon^{-1}|V| - 1} } \right] \right] \cdot \Pr \left[ x(f) \leq \frac{\delta}{b} \right]
\].
≤ Pr \left[ b \in \left[ \frac{w(f)}{12e^{-1}|V| + 2}, \frac{w(f)}{12e^{-1}|V| - 1} \right] \right] \\
\cdot \Pr \left[ x(f) \leq \frac{\delta}{b} \right] \\
\leq \frac{6}{12e^{-1}|V| + 2} \cdot \Pr \left[ x(f) \leq \frac{\delta}{b} \right] \\
= \frac{6}{12e^{-1}|V| + 2} \cdot \frac{\epsilon \cdot \opt_{G,w}(s,t)}{\delta \cdot 12|V|} \\
\leq \frac{6}{12e^{-1}|V|} \cdot \epsilon \cdot \opt_{G,w}(s,t) \leq \opt_{G,w}(s,t),

where the second inequality is from

Pr \left[ b \in \left[ \frac{w(f)}{12e^{-1}|V| + 2}, \frac{w(f)}{12e^{-1}|V| - 1} \right] \right] \\
= \frac{12e^{-1}|V|}{\opt_{G,w}(s,t)} \cdot \max \left( 0, \min \left( \frac{w(f)}{12e^{-1}|V| - 1}, \frac{\opt_{G,w}(s,t)}{6e^{-1}|V|} \right) - \frac{w(f)}{12e^{-1}|V| + 2} \right) \\
\leq \frac{12e^{-1}|V|}{\opt_{G,w}(s,t)} \cdot \max \left( 0, \min \left( \frac{w(f)}{12e^{-1}|V| - 1}, \frac{\opt_{G,w}(s,t)}{6e^{-1}|V|} \right) \cdot \frac{12e^{-1}|V| - 1}{12e^{-1}|V| + 2} \right) \\
\leq \frac{12e^{-1}|V|}{\opt_{G,w}(s,t)} \cdot \max \left( 0, \frac{\opt_{G,w}(s,t)}{6e^{-1}|V|} \cdot \frac{12e^{-1}|V| + 2}{12e^{-1}|V| + 2} \right) \\
= \frac{12e^{-1}|V|}{\opt_{G,w}(s,t)} \cdot \frac{\opt_{G,w}(s,t)}{6e^{-1}|V|} \cdot \frac{3}{12e^{-1}|V| + 2} \\
= \frac{12e^{-1}|V| + 2}{6} \cdot \frac{3}{12e^{-1}|V| + 2} \\
= \frac{12e^{-1}|V| + 2}{6} \cdot \frac{3}{12e^{-1}|V| + 2} \\
= \frac{12e^{-1}|V| + 2}{6} \cdot \frac{3}{12e^{-1}|V| + 2} \square

Finally, we consider Case (iii).

**Lemma IV.8.** For \( b \in \mathbb{R}_{\geq 0} \), let \( E_b \) be the event where Case (iii) occurs with \( b = b^{\delta f} = b \). Then, we have

\[
\mathbb{E}_{((b,x),(b^{\delta f}, x^{\delta f})) \sim \mathcal{D}} \left[ \text{EM}_{\text{w}}(\text{LipSP}(G, w, s, t, \epsilon)) \right] \\
\leq O \left( \frac{\log^3(|V|/|E|)}{\epsilon} \right) \cdot b + \delta.
\]

**Proof.** Let \( Q_1, Q_2 \) be the two paths in \( \hat{G}^{\delta f} \) corresponding to \( f \). Because \( \hat{w}(f) \geq 2 \) (and thus \( \hat{w}^{\delta f}(f) \geq 3 \)), we can take edges \( e_1, e_2 \) on \( Q_1, Q_2 \), respectively, such that both of the endpoints have indegree and outdegree 1, i.e., we can assume that \( e_1 \) and \( e_2 \) are contractible. Therefore by Lemma IV.3, we have

\[
\text{EM}_u \left( \text{DiSP}(\hat{G}^{\delta f}, s, t, \epsilon/4), \text{DiSP}(\hat{G}^{\delta f}/e_1, s, t, \epsilon/4) \right) \leq O \left( \frac{\log^3(\hat{V})}{\epsilon} \right).
\]

Clearly, \( e_2 \) is still contractible in \( \hat{G}/e_1 \). Thus, we have

\[
\text{EM}_u \left( \text{DiSP}(\hat{G}^{\delta f}, s, t, \epsilon/4), \text{DiSP}(\hat{G}, s, t, \epsilon/4) \right)
\]

\[
= \text{EM}_u \left( \text{DiSP}(\hat{G}^{\delta f}, s, t, \epsilon/4), \text{DiSP}((\hat{G}^{\delta f}/e_1)/e_2, s, t, \epsilon/4) \right)
\]

\[
\leq \text{EM}_u \left( \text{DiSP}(\hat{G}^{\delta f}, s, t, \epsilon/4), \text{DiSP}(\hat{G}^{\delta f}/e_1, s, t, \epsilon/4) \right)
\]

\[
+ \text{EM}_u \left( \text{DiSP}(\hat{G}^{\delta f}/e_1, s, t, \epsilon/4), \text{DiSP}(\hat{G}^{\delta f}/e_1)/e_2, s, t, \epsilon/4) \right)
\]

\[
\leq O \left( \frac{\log^3(\hat{V})}{\epsilon} \right).
\]

Let \( b \) and \( x, x^{\delta f} \in [0, 1]^E \) be such that \( ((b, x), (b^{\delta f}, x^{\delta f})) \) is in the support of \( \mathcal{D} \) and Case (iii) occurs when \( b = b^{\delta f} = b, x = x^{\delta f} = x^{\delta f} \). Let \( \hat{G} \) (resp., \( \hat{G}^{\delta f} \)) be the graph constructed from \( (G, w) \) (resp., \( (G, w + \delta 1_f) \)) using \( (b, x) \) (resp., \( (b, x^{\delta f}) \)). Let \( (P^{\delta f}, \hat{P}) \) be a pair sampled from the coupling \( \hat{\mathcal{D}}_{\hat{G}^{\delta f}} \) that attains

\[
\text{EM}_u \left( \text{DiSP}(\hat{G}^{\delta f}, s, t, \epsilon/4), \text{DiSP}(\hat{G}, s, t, \epsilon/4) \right)
\]

\[
= \text{EM}_u \left( \text{DiSP}(\hat{G}^{\delta f}, s, t, \epsilon/4), \text{DiSP}(\hat{G}^{\delta f}/e_1/e_2, s, t, \epsilon/4) \right)
\]

\[
\leq \text{EM}_u \left( \text{DiSP}(\hat{G}^{\delta f}, s, t, \epsilon/4), \text{DiSP}(\hat{G}^{\delta f}/e_1, s, t, \epsilon/4) \right)
\]

\[
+ \text{EM}_u \left( \text{DiSP}(\hat{G}^{\delta f}/e_1, s, t, \epsilon/4), \text{DiSP}(\hat{G}^{\delta f}/e_1)/e_2, s, t, \epsilon/4) \right)
\]

\[
\leq O \left( \frac{\log^3(\hat{V})}{\epsilon} \right).
\]

Let \( P \) and \( P^{\delta f} \) be the walks in \( G \) corresponding to \( P \) and \( \hat{P}^{\delta f} \), respectively. Then, we have

\[
\left\| \sum_{e \in E(P)} w(e)1_e - \sum_{e \in E(P^{\delta f})} (w(e) + \delta 1_{e=\delta f})1_e \right\|_1 \leq \left\| \sum_{e \in E(P)} w(e)1_e - \sum_{e \in E(P^{\delta f})} w(e)1_e \right\|_1 + \delta
\]

\[
= \sum_{e \in E(P) \triangle E(P^{\delta f})} w(e) + \delta
\]

\[
\leq \sum_{e \in E(P) \triangle E(P^{\delta f})} (l(e) + 1)b + \delta
\]

\[
\leq \sum_{e \in E(P) \triangle E(P^{\delta f})} \hat{w}(e)b + \delta
\]

\[
= \left| E(\hat{P}) \triangle E(\hat{P}^{\delta f}) \right| \cdot b + \delta.
\]

Therefore, we have

\[
\mathbb{E}_{((b,x),(b^{\delta f}, x^{\delta f})) \sim \mathcal{D}} \left[ \text{EM}_{\text{w}}(\text{LipSP}(G, w, s, t, \epsilon)) \right]
\]

\[
\leq O \left( \frac{\log^3(|V|/|E|)}{\epsilon} \right) \cdot b + \delta.
\]
\begin{equation}
\begin{aligned}
\text{LipSP}(G, w + \delta 1_f, s, t, \epsilon) &\mid E_b \\
\leq \left( \frac{O(\log^3|V|E|)}{\epsilon} \cdot b + \delta \right) \cdot \frac{\delta}{b} \\
\leq O(\delta) + O(\delta) + O\left( \frac{\delta \log^3|V|E|}{\epsilon} \right) + \delta^2 \\
\leq O\left( \frac{\delta \log^3|V|E|}{\epsilon} \right),
\end{aligned}
\end{equation}

where the last inequality is from \( \delta \leq \frac{\epsilon \text{opt}_G(s,t)}{12|V|} \leq b. \)

**Proof of Theorem IV.1.** Combining Lemmas IV.4 and IV.9 yields the claim. \( \square \)

1) **Proof of Lemma IV.5:** To prove Lemma IV.5, we first present several properties of the coupling \( D. \)

**Lemma IV.10.** In the coupling \( D, \) we have \( \hat{w}(\epsilon) = \hat{w}^\delta(\epsilon) \) for any \( \epsilon \in E \setminus \{f\}. \)

**Proof.** The claim holds because \( x(\epsilon) = x^\delta f(\epsilon) \) and \( (l(\epsilon)+1)b-w(\epsilon) = (l^\delta f(\epsilon)+1)b-w(\epsilon) \) holds if \( \epsilon \neq f. \) \( \square \)

**Lemma IV.11.** In the coupling \( D, \) we have \( \hat{w}(f) = \hat{w}^\delta f(f) \) if \( x(f) > \delta. \)

**Proof.** Note that \( l^\delta f(\epsilon) \) is equal to either \( l(\epsilon) \) or \( l(\epsilon) + 1 \) because \( \delta \leq \frac{\epsilon \text{opt}_G(s,t)}{12|V|}. \)

Suppose \( l^\delta f(f) = l(f). \) Then, we have

\( \hat{w}(f) = l(f) + 2 \iff x(f) \leq \frac{(l(f)+1)b-w(f)}{b} \)

\( \iff x(f) - \frac{\delta}{b} \leq \frac{(l(f)+1)b-w(f) - \delta}{b} \)

\( \iff x^\delta f(f) \leq \frac{(l(f)+1)b-(w(f)+\delta)}{b} \)

\( \iff \hat{w}^\delta f(f) = l(f) + 2. \)

Suppose \( l^\delta f(f) = l(f) + 1. \) Then from the definitions of \( l(f) \) and \( l^\delta f(f), \) we have

\( \frac{w(f)}{b} < l(f) + 1 \leq \frac{w(f) + \delta}{b}. \)

It follows that

\( x(f) > \frac{\delta}{b} = \frac{w(f) + \delta - w(f)}{b} \geq l(f) + 1 - \frac{w(f)}{b} \)

\( = \frac{(l(f)+1)b-w(f)}{b}, \)

which implies \( \hat{w}(f) = l(f) + 3. \) Moreover, we have

\( x^\delta f(f) = x(f) - \frac{\delta}{b} \leq 1 - \frac{\delta}{b} \)

\( \leq \frac{(l(f)+1)b-(w(f)+\delta)}{b} \)

\( = \frac{(l^\delta f(f)+1)b-(w(f)+\delta)}{b}. \)
where the last inequality is from \((l(f) + 1)b \geq w(f)\). Therefore \(\hat{\omega}^\delta(f) = l(f) + 3 = \hat{\omega}(f)\), and the claim holds.

\[ \text{Lemma IV.12. In the coupling } \mathcal{D}, \text{ we have } \hat{\omega}(f) = \hat{\omega}^\delta(f) - 1 \text{ if } x(f) \leq \frac{\delta}{b}. \]

\[ \text{Proof. Suppose } l^\delta(f) = l(f). \text{ Then, we have} \]
\[ x(f) \leq \frac{\delta}{b} = \frac{w(e) + \delta - w(e)}{b} < \frac{l(f) + 1 - w(e)}{b} = \frac{(l(f) + 1)b - w(e)}{b}. \]

Thus, \(\hat{\omega}(f) = l(f) + 2\). Moreover, we have
\[ x^\delta(f) = x(f) - \frac{\delta}{b} + 1 \geq 1 - \frac{\delta}{b} \]
\[ \geq \frac{l(f)b - w(f)}{b} + 1 - \frac{\delta}{b} \]
\[ = \frac{(l(f) + 1)b - (w(f) + \delta)}{b}, \]

where the last inequality is from \(l(f)b \leq w(f)\). Therefore \(\hat{\omega}^\delta(f) = l(f) + 3 = \hat{\omega}(f) + 1\).

Suppose \(l^\delta(f) = l(f) + 1\). Then, we have
\[ \hat{\omega}(f) = l(f) + 2 \]
\[ \iff x(f) \leq \frac{(l(f) + 1)b - w(f)}{b} \]
\[ \iff x(f) - \frac{\delta}{b} + 1 \leq \frac{(l(f) + 1)b - w(f)}{b} - \frac{\delta}{b} + 1 \]
\[ \iff x^\delta(f) \leq \frac{(l(f) + 2)b - (w(f) + \delta)}{b} \]
\[ \iff x^\delta(f) \leq \frac{(l(f) + 2)b - (w(f) + \delta)}{b}. \]

Therefore, the lemma is proved.

\[ \text{Proof of Lemma IV.5. Combining the three lemmas above yields the lemma.} \]

V. MAXIMUM WEIGHT MATCHING

In this section, we present a Lipschitz continuous algorithm for the maximum weight matching problem and prove the following:

**Theorem V.1.** For any \(\epsilon \in (0, 1/8)\), there exists a polynomial-time \((1/8 - \epsilon)\)-approximation algorithm with Lipschitz constant \(O(\epsilon^{-1})\).

Let \(\alpha > 2\). Our algorithm is almost identical to the one used in the algorithm for the maximum weight matching problem with a sensitivity bound [30]. We classify edges according to their weights using the geometric sequence \(b \cdot \alpha^i\), where \(b\) is sampled uniformly from \([1, \alpha]\), and then iteratively apply the randomized greedy to the edge sets from the heaviest one to the lightest one to construct the output matching. The only difference from the algorithm used in [30] is that we use \(b\) to perturb the thresholds.

The approximability analysis is identical to that presented in [30].

**Lemma V.2.** The approximation ratio of Algorithm 4 is (at least) \(\frac{1}{4} \frac{\epsilon}{\omega} \).

Now, we analyze the Lipschitz continuity. By Lemma I.7, it suffices to bound

\[ EM_w (LipMWM(G, w, \alpha), w), \]
\[ (LipMWM(G, w - \delta 1_f, \alpha), w - \delta 1_f)). \]

We use \(b^\delta\) and \(M^\delta\) to denote the random variables \(b\) and \(M\), respectively, used in LipMWM\((G, w - \delta 1_f, \alpha)\). Let \(i_b\) be the (unique) integer such that \(b \cdot \alpha^{i_b} \leq w(f) < b \cdot \alpha^{i_b+1}\). The following is immediate from the algorithm.

**Lemma V.3.** Let \(b \in [1, \alpha]\). If \(b \cdot \alpha^{i_b} \leq w(f) - \delta\), then we have

\[ EM_w (LipMWM(G, w, \alpha) | b = b), w), \]
\[ (LipMWM(G, w - \delta 1_f, \alpha) | b^\delta = b), w - \delta 1_f)) = 0. \]

Let us consider the case where the condition of Lemma V.3 does not hold. Then, we have the following. We omit the proof because a similar argument was presented in [30].

**Lemma V.4.** Let \(b \in [1, \alpha]\). Suppose that \(b = b^\delta = b\) and \(w(f) - \delta < b \cdot \alpha^{i_b}\). Then, for any \(i > i_b\), the distributions of \(E_i \cap M_i\) and \(E_i^\delta \cap M^\delta\) are exactly the same, where we write \(E_i\) and \(E_i^\delta\) instead of \(E_i\) and \(E_i^\delta\) respectively.
and $E_{f}^{\delta f}$, respectively, because they are deterministically constructed from $b$. Additionally, there is a coupling $\mathcal{M}$ between $\mathcal{M}$ and $\mathcal{M}^{\delta f}$ such that

$$\mathbb{E}_{(M,M^{\delta f}) \sim \mathcal{M}} \left[ \left( (E_{i} \cap M) \Delta (E_{i}^{\delta f} \cap M^{\delta f}) \right) \right] \leq 2^{i_{b} - i + 1}$$

holds for all $i \leq i_{b}$.

Now we have the following.

**Lemma V.5.** Let $b \in [1, \alpha]$ satisfy $w(f) - \delta < b \cdot \alpha^{i_{b}}$.

Then, we have

$$\mathbb{E}_{w} \left( (\text{LipMWM}(G, \alpha) \mid b = b), (\text{LipMWM}(G - \delta f, \alpha) \mid b^{\delta f} = b) \right) \leq \frac{8\alpha}{\alpha - 2} \cdot w(f).$$

**Proof.** We have

$$\mathbb{E}_{w} \left( (\text{LipMWM}(G, \alpha) \mid b = b), (\text{LipMWM}(G - \delta f, \alpha) \mid b^{\delta f} = b) \right) \leq \sum_{i = -\infty}^{i_{b}} \left( \min_{M, M^{\delta f} \sim \mathcal{M}} \mathbb{E}_{(M,M^{\delta f}) \sim \mathcal{M}} \left[ \left( (E_{i} \cap M) \Delta (E_{i}^{\delta f} \cap M^{\delta f}) \right) \right] \cdot b \cdot \alpha^{i+1} \right) \leq \sum_{i = -\infty}^{i_{b}} \left( (2^{i_{b} - i + 1} \cdot b \cdot \alpha^{i+1}) \right) \leq 4b \cdot \frac{1}{1 - \frac{2}{\alpha}} \cdot \alpha^{i_{b}} \leq \frac{8\alpha}{\alpha - 2} \cdot w(f).$$

Now, we have the following bound on the Lipschitz constant.

**Lemma V.6.** We have

$$\mathbb{E}_{w} \left( (\text{LipMWM}(G, w, \alpha), w), (\text{LipMWM}(G, w - \delta 1_{f}, \alpha), w - \delta 1_{f}) \right) \leq \frac{12\alpha^{3}}{\alpha - 2} \cdot \delta.$$
connecting them via two edges $e_1$ and $e_2$ having weights 0 and 1, respectively.

Let $\mathcal{A}$ be an arbitrarily deterministic algorithm for the s-t shortest path problem with a finite approximation ratio. Then, $\mathcal{A}$ must output $e_1$.

Consider a sequence of weighted graphs $G_0 = G, G_1, \ldots, G_k$, where $G_k$ is constructed from $G_{i-1}$ by decreasing and increasing the weights of $e_1$ and $e_2$, respectively, by $1/k$. Note that in $G_k$ the weights of $e_1$ and $e_2$ are 1 and 0, respectively, and hence the algorithm $\mathcal{A}$ must output $e_2$ to achieve a finite approximation ratio. This implies that there exists some $i$ such that $\mathcal{A}$ outputs $e_1$ and $e_2$ on $G_{i-1}$ and $G_i$, respectively. Hence, the Lipschitz constant of $\mathcal{A}$ is at least $(w_{G_{i-1}}(P) + w_{G_i}(Q))/1/k = \Omega(k)$. Since $k$ is arbitrary, the Lipschitz constant of $\mathcal{A}$ must be unbounded.

**Theorem VI.1.** Any (randomized) $(1+\epsilon)$-approximation algorithm for the minimum spanning tree problem has Lipschitz constant $\Omega(\epsilon^{-1})$.

**Proof.** Consider a weighted graph $G$ constructed by introducing two vertices $s$ and $t$ and connecting them via two edges $e_1$ and $e_2$ both with weight 1.

Let $\mathcal{A}$ be an arbitrary $(1+\epsilon)$-approximation algorithm for the minimum spanning tree problem. Without loss of generality, we can assume that $\mathcal{A}$ outputs $e_1$ with probability at least half.

Consider another graph $G'$ obtained from $G$ by replacing the weight of $e_2$ with $1-10\epsilon$. Let $p'_1$ and $p'_2 := 1-p'_1$ be the probabilities that $\mathcal{A}$ on $G'$ outputs the edges $e_1$ and $e_2$, respectively. Then, to achieve $(1+\epsilon)$-approximation, we must have

$$p'_1 \cdot 1 + p'_2 \cdot (1-10\epsilon) \leq (1+\epsilon)(1-10\epsilon),$$

which implies $p'_2 > 9/10 + \epsilon$. It follows that the sensitivity is at least

$$\frac{1}{10\epsilon} \left( \left( \frac{9}{10} + \epsilon \right) \left( 1-10\epsilon \right) - \frac{1}{2} \cdot 1 \right) = \Omega \left( \frac{1}{\epsilon} \right).$$

**Theorem VI.2.** Any (randomized) $(1+\epsilon)$-approximation algorithm for the shortest path problem has Lipschitz constant $\Omega(\epsilon^{-1})$.

**Proof.** We reuse the graph and analysis for the proof of Theorem VI.1 to prove the claim.

**Theorem VI.3.** Any (randomized) $\alpha$-approximation algorithm for the maximum weight matching problem has Lipschitz constant $\Omega(\alpha)$.

$\mathcal{A}$ may output a walk passing through $e_1$ several times, but we can show a (larger) lower bound for this case similarly.

**Proof.** Consider a graph $G = (V, E)$ consisting of two vertices $u, v$ and two edges $e_1, e_2$ connecting them. For $i \in \{1, 2\}$, let $w_i \in \mathbb{R}_{\geq 0}$ be a weight vector such that $w_i(e_i) = 1$ and $w_i(e_{3-i}) = 0$.

Let $\mathcal{A}$ be an $\alpha$-approximation algorithm for the maximum weight matching problem with $\alpha$. Then, $\mathcal{A}(G, w_i)$ must output $e_i$ with probability at least $\alpha$. Hence, we have

$$\frac{EM_{w_i}(\mathcal{A}(G, w_i), (\mathcal{A}(G, w_2), w_2))}{||w_1 - w_2||_1} \geq \frac{2|1 - \alpha|}{2} = \alpha,$$

and the claim holds.

**VII. Pointwise Lipschitzness for Unweighted Mapping**

In this section, we consider pointwise Lipschitz continuity of graph problems with respect to the unweighted mapping. We discuss the minimum spanning tree and maximum weight bipartite matching problems in Sections VII-A and VII-B, respectively.

The following lemma is useful to bound the pointwise Lipschitz constant, which is a counterpart of Lemma I.7. We defer the proof to Appendix A.

**Lemma VII.1.** Let $w \in \mathbb{R}_{\geq 0}^E$ be a weight vector. Suppose that there exist some $c > 0$ and $L > 0$ such that for all $w^* \in \mathbb{R}_{\geq 0}^E$ with $||w - w^*||_1 \leq c$,

$$EM_u(\mathcal{A}(G, w^*), \mathcal{A}(G, w^*+\delta 1_f)) \leq \delta L$$

holds for any $f \in E$ and $0 < \delta \leq c$. Then, the pointwise Lipschitz constant of $\mathcal{A}$ on $G$ at $w \in \mathbb{R}_{\geq 0}^E$ is at most $L$.

**A. Minimum Spanning Tree**

In this section, we prove the following:

**Theorem VII.2.** For any $\epsilon > 0$, there exists a polynomial-time $(1+\epsilon)$-approximation algorithm for the minimum spanning tree problem with the pointwise Lipschitz constant with respect to the unweighted mapping $O(\epsilon^{-1}n/\text{opt})$, where $n$ is the number of vertices in the input graph and opt is the minimum weight of a spanning tree.

Our algorithm, PLipMST, is presented in Algorithm 5. Although the algorithm is almost identical to that of LipMST presented in Section II, the analysis requires some care because we use opt, which can vary depending on the edge weights, in the algorithm. First, we analyze the approximation ratio.
Algorithm 5: (unweighted) Lipschitz-continuous algorithm for the minimum spanning tree problem

1 Procedure PLIPMST\((G = (V, E), w, \epsilon)\)
2 Sample \(b\) uniformly from \(\frac{1}{2} \cdot \frac{\epsilon \cdot \text{opt}(G, w)}{|V| - 1} \cdot \frac{\epsilon \cdot \text{opt}(G, w + \delta 1_f)}{|V| - 1}\);
3 for \(e \in E\) do
4    Sample \(\tilde{w}(e)\) uniformly from \([w(e), w(e) + b]\);
5 return any minimum spanning tree of \(G\) with respect to the weight vector \(\tilde{w}\).

Lemma VII.3. For any graph \(G = (V, E)\), a weight vector \(w \in \mathbb{R}_{\geq 0}^E\), and \(\epsilon > 0\), PLIPMST\((G, w, \epsilon)\) outputs a \((1 + \epsilon)\)-approximate spanning tree.

Proof. Let \(T^*\) be an optimal spanning tree of \(G\), and let \(T\) be the output of PLIPMST\((G, w, \epsilon)\). Let opt and opt\(T\) be the minimum weights of spanning trees in \(G\) with respect to \(w\) and \(\tilde{w}\), respectively. Then, we have

\[
\tilde{\text{opt}} = \sum_{e \in T} \tilde{w}(e) \leq \sum_{e \in T^*} \tilde{w}(e) \leq \sum_{e \in T^*} \left( w(e) + \frac{\epsilon \cdot \text{opt}(G, w)}{|V| - 1} \right) = (1 + \epsilon)\text{opt},
\]

where the first inequality is from the fact that \(T\) is a minimum spanning tree with respect to \(\tilde{w}\), and the claim holds.

Now for a graph \(G = (V, E)\) and \(\epsilon > 0\), we analyze the pointwise Lipschitzness of PLIPMST\((G, \cdot, \epsilon)\) at a weight vector \(w \in \mathbb{R}_{\geq 0}^E\). By Lemma VII.1, it suffices to bound the following for \(f \in E\) at \(\delta > 0\):

\[
\text{EM}_u(\text{PLIPMST}(G, w, \epsilon), \text{PLIPMST}(G, w + \delta 1_f, \epsilon)).
\]

To this end, we consider the coupling \(D\) between \((b, \tilde{w})\) and \((b^{\delta f}, \tilde{w}^{\delta f})\) defined as follows: For each \(b \in \mathbb{R}_{\geq 0}\) and weight vector \(\tilde{w} \in \mathbb{R}_{\geq 0}^E\), we transport the probability mass for \((b, \tilde{w}) = (b, \tilde{w})\) to that for \((b^{\delta f}, \tilde{w}^{\delta f}) = (b, \tilde{w})\) as far as possible in such a way that the probability mass for \(\tilde{w}|_{E\setminus\{f\}} = \tilde{w}|_{E\setminus\{f\}}\) is transported to that for \(\tilde{w}^{\delta f}|_{E\setminus\{f\}} = \tilde{w}^{\delta f}|_{E\setminus\{f\}}\). The remaining mass is transported arbitrarily. For clarity, we denote the minimum weights of spanning trees in \(G\) with respect to weight vectors \(w\) and \(w + \delta 1_f\) by \(\text{opt}(G, w)\) and \(\text{opt}(G, w + \delta 1_f)\), respectively.

Lemma VII.4. We have

\[
\text{TV}(b, b^{\delta f}) \leq \frac{2\delta}{\text{opt}(G, w)}.
\]

Proof. We have

\[
\text{TV}(b, b^{\delta f}) = \frac{\text{Pr}[b^{\delta f} \in \left[ \frac{\epsilon \cdot \text{opt}(G, w)}{|V| - 1} \cdot \frac{\epsilon \cdot \text{opt}(G, w + \delta 1_f)}{|V| - 1} \right]}{\text{opt}(G, w + \delta 1_f)} - \text{opt}(G, w)
\]

\[
\leq \frac{2\delta}{\text{opt}(G, w + \delta 1_f)} \leq \frac{2\delta}{\text{opt}(G, w)}.
\]

Lemma VII.5. For any \(b \in \mathbb{R}_{\geq 0}\), we have

\[
\frac{\text{E}}{(b, \tilde{w}), (b^{\delta f}, \tilde{w}^{\delta f}) \sim D} \left[ \text{TV}(\tilde{w}, \tilde{w}^{\delta f}) \mid b = b^{\delta f} = b \right] \leq \frac{\delta (|V| - 1)}{\epsilon \cdot \text{opt}(G, w)}.
\]

Proof. Recall that if \(b = b^{\delta f} = b\), then the distributions of \(\tilde{w}|_{E\setminus\{f\}} = \tilde{w}|_{E\setminus\{f\}}\) and \(\tilde{w}^{\delta f}|_{E\setminus\{f\}} = \tilde{w}^{\delta f}|_{E\setminus\{f\}}\) are the same. Thus it suffices to bound

\[
\frac{\text{E}}{(b, \tilde{w}), (b^{\delta f}, \tilde{w}^{\delta f}) \sim D} \left[ \text{TV}(\tilde{w}(f), \tilde{w}^{\delta f}(f)) \mid b = b^{\delta f} = b \right] = \frac{\text{Pr}[\tilde{w}(f) \in [w(f), w(f) + \delta]]}{\epsilon \cdot \text{opt}(G, w)} \leq \frac{\delta (|V| - 1)}{\epsilon \cdot \text{opt}(G, w)}.
\]

Lemma VII.6. The pointwise Lipschitz constant of PLIPMST\((G, \cdot, \epsilon)\) at a weight vector \(w \in \mathbb{R}_{\geq 0}^E\) with respect to the unweighted mapping is \(O\left(\frac{1}{\epsilon \cdot \text{opt}(G, w)}\right)\).

Proof. Let \(w^* \in \mathbb{R}_{\geq 0}^E\) be a weight vector with \(||w - w^*|| \leq \text{opt}(G, w)\). By Lemmas II.3, VII.4 and VII.5, we have

\[
\text{EM}_u(\text{MST}(G, w^*, \epsilon), \text{MST}(G, w^* + \delta 1_f, \epsilon))
\]

\[
\leq \frac{2\delta}{\text{opt}(G, w^*) \cdot (|V| - 1) + \frac{\delta (|V| - 1)}{\epsilon \cdot \text{opt}(G, w^*)} \cdot 2}
\]

\[
\leq O\left(\frac{\delta |V|}{\epsilon \cdot \text{opt}(G, w^*)}\right)
\]

By Lemma VII.1, the pointwise Lipschitz constant is bounded by

\[
\min_{\frac{2}{\delta} \leq \text{opt}(G, w)} \text{EM}_u(\text{MST}(G, w^*, \epsilon), \text{MST}(G, w^* + \delta 1_f, \epsilon))
\]

786
Theorem VII.2 follows by combining Lemmas VII.3 and VII.6.

B. Maximum Weight Bipartite Matching

In this section, we present a pointwise Lipschitz continuous algorithm for the maximum weight bipartite matching problem and prove the following:

**Theorem VII.7.** There exists a polynomial-time \((1/2 - \epsilon)\)-approximation algorithm for the maximum weight bipartite matching problem with a pointwise Lipschitz constant with respect to the unweighted mapping \(O(\epsilon^{-1}, n^{3/2}\log m/\text{opt})\), where \(n\) and \(m\) are the numbers of vertices in the left and right parts of the input bipartite graph, respectively, and \(\text{opt}\) is the maximum weight of a matching.

Let \(G = (U \cup V, E)\) be a complete bipartite graph, that is, \(E = U \times V\), and \(w \in \mathbb{R}^{E}_{\geq 0}\) be a weight vector. We assume that vertices in \(U\) and \(V\) are indexed as \(\{1, \ldots, |U|\}\) and \(\{1, \ldots, |V|\}\), respectively. Let us consider the following LP relaxation for the maximum weight bipartite matching problem.

\[
\text{LP}(G, w) := \max_{x \in \{0, 1\}^{E}} \sum_{e \in E} w(e) x_e \\
\quad \text{s.t.} \quad \begin{cases} 
\sum_{j=1}^{|U|} x_{i,j} \leq 1 & (i = 1, \ldots, |U|) \\
\sum_{i=1}^{|V|} x_{i,j} \leq 1 & (j = 1, \ldots, |V|) \\
x_e \geq 0 & (e \in E).
\end{cases}
\]

The downside of this LP is that its optimal solution is not stable for small changes in the input weight vector \(w\). To avoid this issue, we use **entropy regularization** [6]. Specifically, we consider the following optimization problem.

\[
\text{LP}_{\text{ent}}(G, w, B) := \max_{x \in \{0, 1\}^{E}} \sum_{e \in E} w(e) x_e - B \sum_{e \in E} x_e \log x_e \\
\quad \text{s.t.} \quad \begin{cases} 
\sum_{j=1}^{|U|} x_{i,j} \leq 1 & (i = 1, \ldots, |U|) \\
\sum_{i=1}^{|V|} x_{i,j} \leq 1 & (j = 1, \ldots, |V|) \\
x_e \geq 0 & (e \in E),
\end{cases}
\]

where \(B > 0\) is a parameter. The second term (including the negative sign) of the objective function is non-negative; \(\log x_e\) is always non-positive for \(0 \leq x_e \leq 1\).

Our algorithm solves \(\text{LP}_{\text{ent}}(G, w, B)\) for a suitably chosen \(B\) and then rounds the obtained fractional solution \(x\). The rounding process is as follows: First, for each \(i \in \{1, \ldots, |U|\}\) we sample an index \(p(i) \in \{1, \ldots, |V|\}\) using \(\{x_{i,j}\}_{j \in V}\). Intuitively, \(p(i)\) represents the candidate of the vertex in \(|V|\) that will be matched to \(i\). Because there may be multiple indices \(i\) with the same value \(p(i)\), we choose an index \(q(j) \in \{1, \ldots, |U|\} \) with \(p(i) = j\) and include \((q(j), j)\) in the output matching. See Algorithm 6 for details.

1) **Approximation Ratio:** The goal of this section is to show the following:

**Lemma VII.8.** The approximation ratio of PLP\(\text{MWBM}\) is (at least) \(1/2 - \epsilon\).

Throughout this section, we fix a complete bipartite graph \(G = (U \cup V, E = U \times V)\) and a weight vector \(w \in \mathbb{R}^{E}_{\geq 0}\). First, we analyze the loss caused by entropy regularization.

---

**Algorithm 6: Pointwise Lipschitz continuous algorithm for the maximum weight bipartite matching problem**

1. **Procedure** \(\text{PLP}_{\text{MWBM}}(G = (U \cup V, E), w, \epsilon)\)
2. Sample \(B\) uniformly from \(\frac{\epsilon \text{opt}(G, w) - 2\epsilon \text{opt}(G, w)}{|U| \log |V|, |U| \log |V|} \).
3. Let \(x\) be the optimal solution to \(\text{LP}_{\text{ent}}(G, w, B)\).
4. Let \(C_j \leftarrow \emptyset\) for all \(j \in \{1, \ldots, |V|\}\).
5. For \(i \in \{1, \ldots, |U|\}\) do
   6. Set \(p(i) = j\) with probability \(x_{i,j}\) for \(j \in \{1, \ldots, |V|\}\),
   7. \(1\) with the remaining probability.
   8. If \(p(i) \neq 1\) then
      9. Add \(i\) to \(C_{p(i)}\).
   10. \(M \leftarrow \emptyset\).
   11. For \(j \in \{1, \ldots, |V|\}\) do
      12. Sample \(q(j)\) uniformly from \(C_j\);
      13. Add the edge \(\{u_{q(j)}, v_j\}\) to \(M\).
14. Return \(M\).
Lemma VII.9. Let $x \in \mathbb{R}^E$ be the vector as constructed in \textsc{PLipMWBM}(G, w, \epsilon). We have
\[
\sum_{e \in E} w(e)x_e \geq (1-2\epsilon)\text{opt}(G, w).
\]

Proof. Let $x^*$ be the optimal solution to \textsc{LP}(G, w). Then, we have
\[
\text{opt}(G, w) = \sum_{e \in E} w(e)x^*_e \leq \sum_{e \in E} w(e)x^*_e - B \sum_{e \in E} x^*_e \log x^*_e \leq \sum_{e \in E} w(e)x_e - B \sum_{e \in E} x_e \log x_e \leq \sum_{e \in E} w(e)x_e - B|U| \log |V| \leq \sum_{e \in E} w(e)x_e - 2\epsilon \cdot \text{opt}(G, w),
\]
where the first inequality is from the non-negativeness of the regularization term, the second inequality is from the optimality of $x^*$, the third inequality is from
\[
-\sum_{e \in E} x_e \log x_e \leq \sum_{i=1}^{|U|} \log |V| \leq |U| \log |V|,
\]
and the last inequality is from $B \leq 2\epsilon \cdot \text{opt}(G, w)/|U| \log |V|$. \hfill {$\square$}

Next, we analyze the loss caused by rounding. Let $y \in [0, 1]^n$ be a vector, and consider tossing $n$ independent coins with the $t$-th coin turning up with probability $y_i$. Then, the Poisson binomial distribution \cite{13} with respect to $y$, denoted $P(y)$, is the probability distribution of the number of coins turning up. The following shows that this distribution plays an important role in the analysis of the performance of the rounding.

Lemma VII.10. Let $i' \in \{1, \ldots, |U|\}$. Then, \(|C_{P(i')}| - 1\) follows the Poisson binomial distribution with respect to the vector $(x_{1, P(i')}, \ldots, x_{i'-1, P(i'), x_{i'+1, P(i')}, \ldots, x_{|U|, P(i')})$.

Proof. For each $i \in \{1, \ldots, |U|\}$ and $j \in \{1, \ldots, |V|\}$, we have $P_{ij} = x_{i,j}$. Additionally, $P(i)$ (for $i \in \{1, \ldots, |U|\}$) are pairwise independent. Therefore, \(|C_{P(i')}| - 1\) can be regarded as the number of coins turning up when we toss $|U|$ independent coins with turning-up probabilities of $x_{1, P(i')}, \ldots, x_{i'-1, P(i'), x_{i'+1, P(i')}, \ldots, x_{|U|, P(i')}$. \hfill {$\square$}

Let
\[
Y = \inf_{n \geq 1, \sum_{i=1}^n y_i \leq 1} \sum_{k=0}^\infty \frac{1}{k+1} \Pr_{k \sim P(y_1, \ldots, y_n)} [k = k].
\]
Then, the output matching size can be bounded by using $Y$ as can be seen in the following:

Lemma VII.11. Let $x \in \mathbb{R}^E$ be the vector and $M$ be the matching as constructed in \textsc{PLipMWBM}(G, w, \epsilon). Then, we have
\[
\mathbb{E} \left[ \sum_{e \in M} w(e) \right] \geq Y \sum_{e \in E} w(e)x_e.
\]

Proof. We have
\[
\mathbb{E} \left[ \sum_{e \in M} w(e) \right] = \sum_{e \in E} \Pr[e \in M] \cdot \sum_{i=1}^{|U|} \sum_{j=1}^{|V|} \left( \Pr[q(p(i)) = i] \cdot \Pr(x(i, j) = 1) \right) = \sum_{i=1}^{|U|} \sum_{j=1}^{|V|} \left( \Pr[q(p(i)) = i] \cdot \Pr(x(i, j) = 1) \right).
\]

Now we evaluate $Y$.

Lemma VII.12. $Y = \frac{1}{2}$.

Proof. For $y_1, \ldots, y_n \in [0, 1]$ with $\sum_{i=1}^n y_i \leq 1$, let
\[
Y(y_1, \ldots, y_n) = \sum_{k=0}^\infty \frac{1}{k+1} \Pr_{k \sim P(y_1, \ldots, y_n)} [k = k].
\]
Suppose $n \geq 2$. We show that deleting two elements of $(y_1, \ldots, y_n)$, say $y_1$ and $y_2$, and appending their sum does not increase the value. To show this, for $k \in \mathbb{Z}_{\geq 0}$, we let $Q(k) = \Pr_{k \sim P(y_1, \ldots, y_n)} [k = k]$. We have
\[
Y(y_1, y_2, y_3, \ldots, y_n) = \sum_{k=0}^\infty \left( \frac{1}{k+1} \cdot (1-y_1)(1-y_2) \right).
\]
\[ + \frac{1}{k+2} \cdot ((1-y_1)y_2 + y_1(1-y_2)) + \frac{1}{k+3} \cdot y_1y_2 \right) Q_k \right) = \sum_{k=0}^{\infty} \left( \left( \frac{1}{k+1} \cdot y_1y_2 \right) + \sum_{k=0}^{\infty} \left( \left( \frac{1}{k+1} \cdot (1-y_1+y_2) \right) + \frac{1}{k+2} \cdot (y_1+y_2) \right) Q_k \right) \]

Then, we have

\[ Y(y_1, y_2, y_3, \ldots, y_n) - Y(y_1 + y_2, y_3, \ldots, y_n) = \sum_{k=0}^{\infty} \left( \left( \frac{1}{k+1} \cdot y_1y_2 \right) - \frac{1}{k+2} \cdot 2y_1y_2 + \frac{1}{k+3} \cdot y_1y_2 \right) Q_k \]

\[ > y_1y_2 \sum_{k=0}^{\infty} \left( \left( \frac{1}{k+1} - \frac{2}{k+2} + \frac{1}{k+3} \right) Q_k \right) \geq 0. \]

Hence, the minimum of \( Y(y_1, \ldots, y_n) \) is attained when \( n = 1 \). We have

\[ Y(y_1) = (1-y_1) + \frac{1}{2} \cdot y_1 \geq \frac{1}{2}, \]

where the equality holds when \( y_1 = 1 \), and hence the lemma holds.

**Proof of Lemma VII.8.** Combining Lemmas VII.9, VII.11 and VII.12 yields the claim. \( \square \)

2) **Stability of Entropy Regularization:** In this section, we discuss the stability of the solution of the LP with entropy regularization.

For a graph \( G = (V \cup E, E) \), a weight vector \( w \in \mathbb{R}^E \), and \( B > 0 \), let

\[ f(x) = \sum_{e \in E} w(e)x_e - B \sum_{e \in E} x_e \log x_e \]

be the objective function of \( \text{LP}_{\text{ent}} \) with a weight vector \( w \in \mathbb{R}^E_0 \). We first discuss the following property of \( f \).

**Lemma VII.13.** For any \( x, y \in \mathbb{R}^E \) with \( \sum_{j \in V} x_{i,j} \leq 1 \) and \( \sum_{j \in V} y_{i,j} \leq 1 \), we have

\[ f(x) - f(y) \geq \nabla f(x)^T (x - y) + \frac{B}{2} \sum_{i \in U} \left( \sum_{j \in V} |x_{i,j} - y_{i,j}| \right)^2. \]

**Proof.** Because \( f \) is concave, for \( z = x - y \), it suffices to show

\[ z^T H(x)z \leq -B \sum_{i \in U} \left( \sum_{j \in V} |z_{i,j}| \right)^2, \]

where \( H(x) \) is the Hessian of \( f \) at \( x \). Indeed, we have

\[ z^T H(x)z = -B \sum_{i \in U} \sum_{j \in V} \frac{z_{i,j}^2}{x_{i,j}} = -B \sum_{i \in U} \sum_{j \in V} \frac{z_{i,j}^2}{x_{i,j}} \]

\[ \leq -B \sum_{i \in U} \left( \sum_{j \in V} |z_{i,j}| \right)^2. \]

**Lemma VII.14.** Let \( G = (U \cup V, E) \) be a graph, \( w \in \mathbb{R}^E \), a weight vector, \( f \in E \), and \( \delta > 0 \). Let \( x, x^\delta f \in \mathbb{R}^E \) be the optimal solutions to \( \text{LP}_{\text{ent}}(G, w, B) \) and \( \text{LP}_{\text{ent}}(G, w + \delta 1_f, B) \), respectively. Then, we have

\[ ||x - x^\delta f||_1 = O \left( \frac{\delta \sqrt{U}}{B} \right). \]

**Proof.** Let \( f^\delta(x) = \sum_{e \in E \setminus \{f\}} w(e)x_e + (w(f) + \delta)x_f - B \sum_{e \in E} x_e \log x_e \)

be the objective function of \( \text{LP}_{\text{ent}} \) with the weight vector \( w + \delta 1_f \).

By Lemma VII.13, we have

\[ f(x) - f(x^\delta f) \geq \nabla f(x)^T (x - x^\delta f) + \frac{B}{2} \sum_{i \in U} \left( \sum_{j \in V} |x_{i,j} - x^\delta f_{i,j}| \right)^2 \]

\[ \geq \frac{B}{2} \sum_{i \in U} \left( \sum_{j \in V} |x_{i,j} - x^\delta f_{i,j}| \right)^2. \]

Similarly, we have

\[ f^\delta(x^\delta f) - f(x) \geq \frac{B}{2} \sum_{i \in U} \left( \sum_{j \in V} |x_{i,j} - x^\delta f_{i,j}| \right)^2. \]

Also, we have

\[ f(x) - f(x^\delta f) + f^\delta(x^\delta f) - f^\delta(x) = \delta(x^\delta f - x_f). \]
Combining the above three, we have
\[
B \sum_{i \in U} \left( \sum_{j \in V} |x_{i,j} - x_{i,j}^{\delta_f}| \right)^2 \leq \delta(x_f^{\delta_f} - x_f). \tag{12}
\]
In particular, this implies
\[
B(x_f^{\delta_f} - x_f)^2 \leq \delta(x_f^{\delta_f} - x_f),
\]
and hence we have \( |x_f^{\delta_f} - x_f| \leq \delta/B \). By substituting this into (12), we obtain
\[
\sum_{i \in U} \left( \sum_{j \in V} |x_{i,j} - x_{i,j}^{\delta_f}| \right)^2 \leq \frac{\delta^2}{B^2}.
\]
Then, we have
\[
\|x - x^{\delta_f}\|_1 = \sum_{i \in U} \sum_{j \in V} |x_{i,j} - x_{i,j}^{\delta_f}|
\leq \sqrt{\sum_{i \in U} \left( \sum_{j \in V} |x_{i,j} - x_{i,j}^{\delta_f}| \right)^2} \cdot \sqrt{\sum_{i \in U} 1^2} = \frac{\sqrt{|U| \delta}}{B}.
\]

3) **Pointwise Lipschitzness**: The goal of this section is to show the following:

**Lemma VII.15.** For any bipartite graph \( G = (U \cup V, E) \), a weight vector \( w \in \mathbb{R}^E \), \( \delta > 0 \), and \( f \in E \), we have
\[
\begin{align*}
\mathbb{E}_{\mu}(\text{PLIP}(G, w, \epsilon), & \text{PLIP}(G, w + \delta 1_f, \epsilon)) \\
& \leq O \left( \frac{\delta \epsilon^{-1}|U|^2 \log |V|}{\text{opt}(G, w)} \right).
\end{align*}
\]

To bound the earth mover’s distance between the output labelings for \((G, w)\) and \((G, w + \delta 1_f)\), we consider the coupling \( \mathcal{W} \) between \((B, p, q)\) and \((B^{\delta_f}, p^{\delta_f}, q^{\delta_f})\), defined as follows: For each number \( B \in \mathbb{R} \), we transport the probability mass for \( B = B \) to that for \( B^{\delta_f} = B \) as far as possible in such a way that
\[
\mathbb{E}_{((B, p, q), (B^{\delta_f}, p^{\delta_f}, q^{\delta_f})) \sim \mathcal{W}} \left[ \|q - q^{\delta_f}\|_0 \mid B = B^{\delta_f} = B \right],
\]
is minimized, where \( \|q - q^{\delta_f}\|_0 \) is the number of indices \( j \) with \( q(j) \neq q^{\delta_f}(j) \). The remaining mass is transported arbitrarily. Note that \( \|q - q^{\delta_f}\|_0 \) is equal to \( d_u(M, M^{\delta_f}) \), where \( M \) and \( M^{\delta_f} \) are the matchings constructed using \( q \) and \( q^{\delta_f} \), respectively.

First, we evaluate the remaining mass.

**Lemma VII.16.** We have
\[
\text{TV} (B, B^{\delta_f}) \leq \frac{2\delta}{\text{opt}(G, w)}.
\]

**Proof.** We have
\[
\begin{align*}
\text{TV} (B, B^{\delta_f}) &= \text{Pr} \left[ B^{\delta_f} \in \left[ \frac{2\text{opt}(G, w)}{|U| \log |V|}, \frac{2\text{opt}(G, w + \delta 1_f)}{|U| \log |V|} \right] \right] \\
&= 2 \cdot \frac{\text{opt}(G, w + \delta 1_f) - \text{opt}(G, w)}{\text{opt}(G, w + \delta 1_f)} \\
&\leq \frac{2\delta}{\text{opt}(G, w)}.
\end{align*}
\]

Now we evaluate the distance between \( p \) and \( p^{\delta_f} \).

**Lemma VII.17.** Let \( B \in \mathbb{R}_{\geq 0} \) and let \( x, x^{\delta_f} \in \mathbb{R}^E_{\geq 0} \) be the optimal solutions to \( \text{LP}_{\text{ent}}(G, w) \) and \( \text{LP}_{\text{ent}}(G, w + 1_\delta) \), respectively. Then, we have
\[
\text{TV} (B, B^{\delta_f}) = \text{Pr} \left[ B = B^{\delta_f} = B \right] \\
\leq \frac{1}{2} \|x - x^{\delta_f}\|_1.
\]

**Proof.** First, we have
\[
\text{TV} (B, B^{\delta_f}) = \text{Pr} \left[ B = B^{\delta_f} = B \right] \\
\leq \frac{1}{2} \|x - x^{\delta_f}\|_1.
\]

Now we evaluate the distance between \( q \) and \( q^{\delta_f} \).

**Lemma VII.18.** Let \( B \in \mathbb{R}_{\geq 0} \) and let \( x, x^{\delta_f} \in \mathbb{R}^E_{\geq 0} \) be the optimal solutions to \( \text{LP}_{\text{ent}}(G, w) \) and \( \text{LP}_{\text{ent}}(G, w + 1_\delta) \), respectively. Then, we have
\[
\text{TV} (B, B^{\delta_f}) = \text{Pr} \left[ B = B^{\delta_f} = B \right] \\
\leq \|x - x^{\delta_f}\|_1.
\]

**Proof.** Fix \( B, p, p^{\delta_f} \in \mathbb{R}_{\geq 0} \), and let \( (q, q^{\delta_f}) \) be sampled from the marginal distribution of \( \mathcal{W} \) conditioned on \( B = B^{\delta_f} = B, p = p, p = p^{\delta_f} \). Note that under this condition, \( C \) and \( C^{\delta_f} \) are no longer random variables because they are deterministically determined by \( p \) and \( p^{\delta_f} \), respectively. Then, we have
\[
\text{TV} (B, B^{\delta_f}) = \text{Pr} \left[ B = B^{\delta_f} = B \right] \\
\leq \|x - x^{\delta_f}\|_1.
\]
\[ \sum_{i=1}^{|U|} \sum_{j=1}^{|V|} |1_{p(i)=j} - 1_{p^\delta f(i)=j}| \\
+ \sum_{j=1}^{|V|} \left( |C_j| \cdot \left| \frac{1}{|C_j|} - \frac{1}{|C_j^\delta f|} \right| \right) \\
= \|p - p^\delta f\|_0 + \sum_{j=1}^{|V|} \left( |C_j| \cdot \left| \frac{1}{|C_j|} - \frac{1}{|C_j^\delta f|} \right| \right) \\
\leq \|p - p^\delta f\|_0 + \sum_{j=1}^{|V|} \left( |C_j| \cdot \left| \frac{1}{|C_j|} - \frac{1}{|C_j^\delta f|} \right| \right) \\
\leq \|p - p^\delta f\|_0 + \sum_{j=1}^{|V|} \left( C_j^\delta f_j - |C_j| \right) \\
= \|p - p^\delta f\|_0 + \sum_{j=1}^{|V|} \left( 1_{p(i)=j} - 1_{p^\delta f(i)=j} \right) \\
= 2 \cdot \|p - p^\delta f\|_0, \quad (13) \\
\]

where the first inequality is from the fact that \( q(j) \) and \( q^\delta f(j) \) are pairwise independent for \( j \in \{1, \ldots, |V|\} \) and the second inequality is from

\[
\left| \frac{1}{|C_j|} - \frac{1}{|C_j^\delta f|} \right| = \left\{ \begin{array}{ll}
1 & \text{if } 1_{p(i)=j} \neq 1_{p^\delta f(i)=j}, \\
0 & \text{if } 1_{p(i)=j} = 1_{p^\delta f(i)=j} = 1,
\end{array} \right.
\]

and the fact that there are at most \(|C_j|\) indices \( i \) with \( 1_{p(i)=j} = 1_{p^\delta f(i)=j} = 1 \). By integrating (13) over \( p \) and \( p^\delta f \), we obtain

\[
\mathbb{E}_{((B,p,q),(B^\delta f,p^\delta f,q^\delta f)) \sim \mathcal{W}} \left[ \|q - q^\delta f\|_0 \mid B = B^\delta f \right] \\
\leq \mathbb{E}_{((B,p,q),(B^\delta f,p^\delta f,q^\delta f)) \sim \mathcal{W}} \left[ 2 \cdot \|p - p^\delta f\|_0 \mid B = B^\delta f \right] \leq 2 \cdot \|x - x^\delta f\|_1, \\
\]

where the second inequality is from Lemma VII.17. \( \square \)

Now we can bound the pointwise Lipschitz constant as follows.

**Proof of Lemma VII.15.** We have

\[
\mathbb{E}_{\text{PLiPMWB}}(G, w, \epsilon), \\
\mathbb{E}_{\text{PLiPMWB}}(G, w + \delta 1_f, \epsilon) \\
\leq \mathbb{E} \left( B, B^\delta f \right) \cdot |U| + \\
\mathbb{E} \left( ((B,p,q),(B^\delta f,p^\delta f,q^\delta f)) \sim \mathcal{W} \right) \left[ \|q - q^\delta f\|_0 \mid B = B^\delta f \right] \\
\leq \mathbb{E} \left( B, B^\delta f \right) \cdot |U| + \\
\mathbb{E} \left( ((B,p,q),(B^\delta f,p^\delta f,q^\delta f)) \sim \mathcal{W} \right) \left[ 2 \cdot \|p - p^\delta f\|_0 \mid B = B^\delta f \right] \\
\leq \mathbb{E} \left( B, B^\delta f \right) \cdot |U| + \\
\mathbb{E} \left( ((B,p,q),(B^\delta f,p^\delta f,q^\delta f)) \sim \mathcal{W} \right) \left[ 2 \cdot \|x - x^\delta f\|_1 \right], \\
\]

where the second inequality is from Lemma VII.18, and the third inequality is from Lemma VII.14 and VII.16. \( \square \)

**Proof of Theorem VII.7.** The approximability result follows by Lemma VII.8. Let \( w, w^* \in \mathbb{R}_{\geq 0}^Z \) be two weight vectors with \( \text{opt}(G, w) > 0 \) and \( \|w - w^*\| \leq \frac{\text{opt}(G, w)}{2} \).

Then by Lemma VII.15, we have

\[
\mathbb{E}_{\text{PLiPMWB}}(G, w^* + \delta 1_f, \epsilon) \leq \mathbb{E}_{\text{PLiPMWB}}(G, w^*) \\
\leq \mathbb{E}_{\text{PLiPMWB}}(G, w^* + \delta 1_f, \epsilon), \\
\]

Thus, the claim follows by Lemma VII.1. \( \square \)

**VIII. Lipschitz Continuity under Shared Randomness**

In this section, we show that the Lipschitz continuous algorithms we provided in the previous sections can be made Lipschitz continuous under shared randomness. In Sections VIII-A and VIII-B, we provide Lipschitz continuous algorithms for sampling continuous and discrete, respectively, distributions under shared randomness. In Section VIII-C, we explain how we modify the algorithms given in the previous sections to make them Lipschitz continuous under shared randomness.

**A. Sampling from Continuous Uniform Distributions**

In our algorithms, we often sample a value from a uniform distribution over some range that slightly perturbs if we perturb the input weight vector. In this section, we consider sampling procedures such that, even when the range is slightly perturbed, we obtain the same value with a high probability over internal randomness.

Let \( \mathcal{U}([0, 1]) \) and \( \mathcal{U}([0, 1]^{Z \geq 0}) \) be the uniform distributions over \([0, 1]\) and \([0, 1]^{Z \geq 0}\), respectively. Let \( l \) and \( r \) be functions that take weight vectors as arguments and return real values. For \( c \geq 1 \), we say that a sampling process \( \text{SAMPLE}() \) is \( c \)-stable for a pair of functions \( l, r \).
if it samples a value uniformly from \([l(w), r(w)]\) when \(p\) follows the uniform distribution over \([0, 1]^{2}_{\geq 0}\) and
\[
\mathbb{E}_{p \sim \mathcal{U}([0, 1]^{2}_{\geq 0})} \left[ \operatorname{TV} \left( \text{SAMPLE}([l(w), r(w)], p) \right) \right]
\]
\[
= \mathcal{S} \cdot \operatorname{TV} \left( \mathcal{U}([l(w'), r(w')]), \mathcal{U}([l(w'), r(w')]) \right)
\]
holds for all \(w, w' \in \mathbb{R}_{\leq 0}^{E}_{\geq 0}\).

Let us explain how \(c\)-stability can be used to bound the Lipschitz constant in the shared randomness setting using the example of the maximum spanning tree problem. In Algorithm 1, for each edge \(e\), a modified weight \(\tilde{w}(e)\) is sampled from a uniform distribution over \([w(e), (1 + \epsilon)w(e)]\). The analysis of the Lipschitz constant proceeds as follows: For weights \(w\) and \(w + \delta 1_{f}\), a coupling \(\mathcal{W}\) between \(\tilde{w}\) and \(\tilde{w}^{\delta f}\) is defined, and then the Lipschitz constant of the algorithm is proportional to
\[
\Pr_{(\tilde{w}, \tilde{w}^{\delta f}) \sim \mathcal{W}} \left[ \tilde{w}(f) \neq \tilde{w}^{\delta f}(f) \right].
\]
The call to \(\text{SAMPLE}([w(e), (1 + \epsilon)w(e)], p)\) naturally induces a coupling \(\mathcal{W}'\) between \(\tilde{w}\) and \(\tilde{w}^{\delta f}\), that is, we use the same \(p\). The same analysis goes through by considering \(\mathcal{W}'\) instead of \(\mathcal{W}\). Since \(\text{SAMPLE}(\cdot)\) is \(c\)-stable for the pair of functions \((x \mapsto x, x \mapsto (1 + \epsilon)x)\), we have
\[
\mathbb{E}_{(\tilde{w}, \tilde{w}^{\delta f}) \sim \mathcal{W}'} \left[ \tilde{w}(f) \neq \tilde{w}^{\delta f}(f) \right]
\]
\[
\leq c \cdot \operatorname{TV} \left( \mathcal{U}([l(w'), r(w')]), \mathcal{U}([l(w'), r(w')]) \right)
\]
\[
\leq c \cdot \mathbb{P}_{(\tilde{w}, \tilde{w}^{\delta f}) \sim \mathcal{W}'} \left[ \tilde{w}(f) \neq \tilde{w}^{\delta f}(f) \right].
\]
Therefore, in the shared randomness setting, we have obtained an algorithm with a Lipschitz constant that is at most \(c\) times that of the original algorithm.

We consider three algorithms for \(\text{SAMPLE}([l(w), r(w)], p)\). Algorithm 7 will be used when \(l\) and \(r\) are constant functions and do not depend on the weight \(w\). Algorithm 8 will be used when \(l\) and \(r\) are constant functions and do not depend on the weight \(w\). Algorithm 9 will be used when \(l\) and \(r\) are constant functions and do not depend on the weight \(w\).

Algorithm 7: Algorithm for sampling \(x\) uniformly from fixed \([l, r]\)

1. **Procedure** \(\mathcal{U}([l, r], p \in [0, 1])\)

2. return \(l + p \cdot (r - l)\);

Algorithm 8: Algorithm for sampling \(x\) uniformly from \([l, c\cdot l]\)

1. **Procedure** \(\mathcal{U}([l, c\cdot l], p \in [0, 1]^{2}_{\geq 0})\)

2. return \(l + c \cdot (r - l)\);

Algorithm 9: Algorithm for sampling \(x\) uniformly from \([l, l + c]\)

1. **Procedure** \(\mathcal{U}([l, l + c], p \in [0, 1]^{2}_{\geq 0})\)

2. if \(x \geq l\) then return \(x\);

3. if \(x \leq c\cdot l\) then return \(c \cdot l + x\);

The following lemma will be used to make Algorithms 1, 2, and 3 Lipschitz continuous and Algorithms 5 and 6 pointwise Lipschitz continuous even in the shared randomness setting.

Lemma VIII.1. If \(l\) and \(r\) are constant functions, \(\mathcal{U}([l, r], p)\) is \(1\)-stable for \(l\) and \(r\).

The following lemma will be used to make Algorithms 1, 2, and 3 Lipschitz continuous and Algorithms 5 and 6 pointwise Lipschitz continuous even in the shared randomness setting.

Lemma VIII.2. Let \(c > 1\) and suppose \(r(w) = c\cdot l(w)\) holds for all \(w\). Then, \(\mathcal{U}([l(w), c\cdot l(w)], p)\) is \((1 + c)\)-stable for \(l\) and \(r\).

Proof. It is clear that the distribution of \(\mathcal{U}([l(w), c\cdot l(w)], p)\) is uniform over \([l(w), c\cdot l(w)]\) when \(p \sim \mathcal{U}([0, 1]^{2}_{\geq 0})\). For weight vectors \(w\) and \(w'\), let us prove
\[
\mathbb{E}_{p \sim \mathcal{U}([0, 1]^{2}_{\geq 0})} \left[ \operatorname{TV} \left( \mathcal{U}([l(w), c\cdot l(w)], p) \right) \right]
\]
\[
= \mathcal{S} \cdot \mathcal{O}\left(1 + c\cdot l(w)\right)\mathcal{O}\left(1 + c\cdot l(w)\right).
\]
UniformFixedRatio(\([l(w'), cl(w')], p)\])
\leq \frac{1}{c} \cdot TV(\mathcal{U}([l(w), cl(w)]), \mathcal{U}([l(w'), cl(w')])).

Without loss of generality, we can assume \(l(w) \leq l(w')\). We can also assume \(l(w') < cl(w)\) because otherwise the right hand side is equal to \(1 + c\) and the lemma trivially holds.

Let \(k\) and \(k'\) be the values of \(k\) chosen in UniformFixedRatio(\([l(w), cl(w)], p\)) and UniformFixedRatio(\([l(w'), cl(w')], p\)), respectively.

Assume \(k = k'\). Then, it clearly holds

\[TV(\text{UniformFixedRatio}(\[l(w), cl(w)], p), \text{UniformFixedRatio}(\[l(w'), cl(w'), p]\)) = TV(\mathcal{U}([l(w), cl(w)]), \mathcal{U}([l(w'), cl(w')])).\]

Let us evaluate the probability \(k \neq k'\) happens if \(p\) is sampled from \(\mathcal{U}([0, 1]^{\geq 0})\). From \(l(w) \leq l(w')\), we have \(c^k \leq \frac{l(w)}{b} < c^{k+1} \leq \frac{l(w')}{b} < \frac{cl(w)}{b} \leq c^{k+2}\). This can happen only when \(k = \lfloor \log_l \frac{l(w)}{b} \rfloor = k\) or \(k = \hat{k} - 1\) because of \(b \in [1, c]\). Furthermore, \(\frac{l(w)}{b} < c^k \leq \frac{l(w')}{b}\) and \(\frac{l(w)}{b} < \frac{l(w')}{b} = \frac{l(w')}{b}\) cannot happen at the same time because \(l(w') < cl(w)\). If \(l(w') < c^{k+1}\), we have

\[
\Pr_{p \sim \mathcal{U}([0, 1]^{\geq 0})}[k \neq k'] = \Pr_{p \sim \mathcal{U}([0, 1]^{\geq 0})}\left[\frac{l(w)}{b} < c^k \leq \frac{l(w')}{b} \lor \frac{l(w)}{b} < c^{k+1} \leq \frac{l(w')}{b}\right]
= \Pr_{p \sim \mathcal{U}([0, 1]^{\geq 0})}\left[\frac{l(w)}{b} \leq \frac{l(w')}{b}\right] = \Pr_{p \sim \mathcal{U}([0, 1]^{\geq 0})}\left[\frac{l(w)}{b} \leq \frac{l(w')}{b}\right]
= \Pr_{p \sim \mathcal{U}([0, 1]^{\geq 0})}\left[\frac{l(w)}{b} \leq \frac{l(w')}{b}\right]
= \frac{l(w') - l(w)}{c^k} \leq \frac{l(w') - l(w)}{c} \leq \frac{l(w') - l(w)}{c},
\]

where the third equality is from \(\frac{l(w')}{c} \in [l(w'), c^{k+1}]\). Otherwise, we have

\[
\Pr_{p \sim \mathcal{U}([0, 1]^{\geq 0})}[k \neq k'] = \Pr_{p \sim \mathcal{U}([0, 1]^{\geq 0})}\left[\frac{l(w)}{b} < c^k \leq \frac{l(w')}{b} \lor \frac{l(w)}{b} < c^{k+1} \leq \frac{l(w')}{b}\right]
= \Pr_{p \sim \mathcal{U}([0, 1]^{\geq 0})}\left[\frac{l(w)}{b} < c^k \leq \frac{l(w')}{b}\right] = \Pr_{p \sim \mathcal{U}([0, 1]^{\geq 0})}\left[\frac{l(w)}{b} < c^k \leq \frac{l(w')}{b}\right]
\leq \frac{1}{c} \cdot \frac{l(w') - l(w)}{c^k} \leq \frac{1}{c} \cdot \frac{l(w') - l(w)}{c},
\]

where the third equality is from \(\frac{l(w')}{c} \in [l(w'), c^{k+1}]\). Furthermore, we have

\[
TV(\mathcal{U}([l(w), cl(w)]), \mathcal{U}([l(w'), cl(w')]))) = \frac{c}{c - 1} \cdot \frac{l(w') - l(w)}{l(w') - l(w)} = \frac{c}{c - 1} \cdot \frac{l(w') - l(w)}{l(w')}
\geq \frac{1}{c} \cdot \Pr_{p \sim \mathcal{U}([0, 1]^{\geq 0})}[k \neq k']
\]

Thus, we have

\[
\mathbb{E}_{p \sim \mathcal{U}([0, 1]^{\geq 0})}[TV(\text{UniformFixedRatio}(\[l(w), cl(w)], p), \text{UniformFixedRatio}(\[l(w'), cl(w'), p]\))]
\leq TV(\mathcal{U}([l(w), cl(w)]), \mathcal{U}([l(w'), cl(w')]))) + 1 \cdot \Pr_{p \sim \mathcal{U}([0, 1]^{\geq 0})}[k \neq k']
\leq (1 + c) \cdot TV(\mathcal{U}([l(w), cl(w)]), \mathcal{U}([l(w'), cl(w')]))) \quad \Box
\]

The following will be used to make Algorithm 6 point-wise Lipschitz continuous even in the shard randomness setting.

**Lemma VIII.3.** Let \(c > 0\) and suppose \(r(w) = l(w) + c\) holds for all \(w\). Then, UniformFixedWidth([l(w), r(w) = l(w) + c], p) is 1-uniform for l and r.

**Proof.** It is clear that the distribution of UniformFixedWidth([l(w), r(w) = l(w) + c], p) is uniform distribution over \([l(w), l(w) + c]\) when \(p \sim \mathcal{U}([0, 1]^{\geq 0})\). Let \(w\) and \(w'\) be weight vectors. Without loss of generality, we can assume \(l(w) \leq l(w')\). We can also assume \(l(w') \leq l(w) + c\) because otherwise \(TV(\mathcal{U}([l(w), l(w) + c]), \mathcal{U}([l(w'), l(w') + c])) = 1\) and the lemma trivially holds. Thus, we can assume \(l(w') = \frac{l(w')}{c}\) is equal to either \(\frac{l(w)}{c}\) or \(\frac{l(w)}{c}\) + 1.

Assume \(\frac{l(w')}{c} = \frac{l(w)}{c}\). Then, UniformFixedWidth([l(w), l(w) + c], p) and UniformFixedWidth([l(w'), l(w') + c], p) will be distinct only when \((l(w) \mod c) < x \leq (l(w') \mod c)\). Thus we have

\[
\mathbb{E}_{p \sim \mathcal{U}([0, 1]^{\geq 0})}[TV(\text{UniformFixedWidth}(\[l(w), l(w) + c], p),
\text{UniformFixedWidth}(\[l(w'), l(w') + c], p))]
\]

793
\[ \text{UNIFORMFIXEDWIDTH}([l(w'), l(w') + c], p) \]
\[ = \frac{l(w') - l(w)}{c} \]
\[ = \text{TV} \left( \mathcal{U}([l(w), l(w) + c]), \mathcal{U}([l(w'), l(w') + c]) \right). \]

Assume \[ \left\lceil \frac{l(w')}{c} \right\rceil = \left\lceil \frac{l(w)}{c} \right\rceil + 1 \]. Then, \( \text{UNIFORMFIXEDWIDTH}([l(w), l(w) + c], p) \) and \( \text{UNIFORMFIXEDWIDTH}([l(w'), l(w') + c], p) \) will be distinct only when \( x > (l(w) \mod c) \) or \( x \leq (l(w') \mod c) \). Thus we have

\[ \mathbb{E}_{p \sim \mathcal{U}([0, 1]^{\geq 0})} \left[ \text{TV} \left( \text{UNIFORMFIXEDWIDTH}([l(w), l(w) + c], p), \text{UNIFORMFIXEDWIDTH}([l(w'), l(w') + c], p) \right) \right] \]
\[ = \left( l(w') \mod c \right) + c - \left( l(w) \mod c \right) \]
\[ = \frac{l(w') - l(w)}{c} \]
\[ = \text{TV} \left( \mathcal{U}([l(w), l(w) + c]), \mathcal{U}([l(w'), l(w') + c]) \right), \]

where the second equality is from \( \frac{l(w')}{c} = \frac{l(w)}{c} + 1 \). Therefore the lemma is proved.

**B. Sampling from Discrete Distributions**

In our algorithms, we often sample a value from a discrete universe. Let \( \mathcal{X} \) be a universe and \( x \in [0, 1]^\mathcal{X} \) be a probability vector, i.e., \( \sum_{x \in \mathcal{X}} x_v = 1 \). Then, we say that a sampling process \( \text{SAMPLE} (\mathcal{X}, x, p) \) is stable if it samples a value from \( x_v \) with probability \( x_v \) when \( p \sim \mathcal{U}([0, 1]^{\geq 0}) \) and for two probability vectors \( x \) and \( x' \), it holds that

\[ \Pr_{p \sim \mathcal{U}(\mathcal{X}^{\geq 0})} \left[ \text{SAMPLE} (\mathcal{X}, x, p) \neq \text{SAMPLE} (\mathcal{X}, x', p) \right] \]
\[ = \text{TV} (x, x'). \]

Here, we abuse notation \( \text{TV} (x, x') \) to represent the total variation distance between the probability distribution for sampling an element \( v \) from \( \mathcal{X} \) with probability \( x_v \) and the probability distribution for sampling an element \( v \) from \( \mathcal{X} \) with probability \( x'_v \).

We denote the uniform distribution over \( \mathcal{X} \) by \( \mathcal{U}(\mathcal{X}) \). Algorithm 10 is the algorithm we consider, and it is stable as shown in the following.

**Lemma VIII.4.** \( \text{UNIFORMDISCRETE}(U, \mathcal{X}, x, p \in \mathcal{X}^{\geq 0}) \) is stable.

**Proof.** It is clear that \( \text{UNIFORMDISCRETE}(\mathcal{X}, x \in [0, 1]^\mathcal{X}, p \in [0, 1]^{\geq 0}) \) outputs an element \( v \in \mathcal{X} \) with probability \( x_v \) when \( p \sim \mathcal{U}([0, 1]^{\geq 0}) \). Let \( x, x' \in [0, 1]^\mathcal{X} \) be probability vectors such that \( \sum_{v \in \mathcal{X}} x_v = \sum_{v \in \mathcal{X}} x'_v = 1 \). Let \( i^* \) denote the value of \( i \) when \( \text{UNIFORMDISCRETE}(\mathcal{X}, x \in [0, 1]^\mathcal{X}, p \in [0, 1]^{\geq 0}) \) terminates. Then,

\[ \Pr_{p \sim \mathcal{U}(\mathcal{X}^{\geq 0})} \left[ \text{UNIFORMDISCRETE}(\mathcal{X}, x, p) \neq \text{UNIFORMDISCRETE}(\mathcal{X}, x', p) \right] \]
\[ = \text{TV} (x, x'). \]

Thus, we have

\[ \Pr_{p \sim \mathcal{U}(\mathcal{X}^{\geq 0})} \left[ \text{UNIFORMDISCRETE}(\mathcal{X}, x, p) \right] \neq \text{UNIFORMDISCRETE}(\mathcal{X}, x', p) = \text{TV} (x, x'). \]

When \( p \sim [0, 1]^{\geq 0} \), it can be verified that Algorithm 10 runs in expected time \( O(|\mathcal{X}|) \), as each step of the loop terminates with probability \( \sum_{v \in \mathcal{X}} \left( \frac{1}{|\mathcal{X}|} x_v \right) = \frac{1}{|\mathcal{X}|} \).

**C. Modifications to Ensure Lipschitz Continuity under Shared Randomness**

1) **Algorithm 1:** By modifying Algorithm 1 to use \( \text{SAMPLEFIXEDRATIO}([w(e), (1 + \epsilon)w(e)], p) \) to sample \( \tilde{w}(e) \) at Line 3, it can be converted into a Lipschitz continuous algorithm in the shared randomness setting. Although the right hand side of the statement in Lemma II.4 becomes \( 2 + \epsilon \) times as large, the proof of Lemma II.5 still works in the same way. Therefore, an algorithm achieving an approximation ratio of \( 1 + \epsilon \) and a Lipschitz constant of \( O(\epsilon^{-1}) \) is obtained in the shared randomness setting.

2) **Algorithm 3:** To make Algorithm 3 Lipschitz continuous in the shared randomness setting, first, we use \( \text{UNIFORMFIXEDRATIO} \) to sample \( b \) at Line 4. Although the right hand side of the statement in Lemma IV.6 becomes three times as large, the proof of Lemma IV.9 still works in the same way. In Line 8 of Algorithm 3, \( x \)

| Algorithm 10: Algorithm to sample \( v \) from a universe \( \mathcal{X} \) with probability \( x_v \) |
| --- |
| **Procedure** \( \text{UNIFORMDISCRETE}(\mathcal{X}, x, p) \) |
| 1. Procedure \( \text{UNIFORMDISCRETE}(\mathcal{X}, x, p) \) |
| 2. \textbf{for} \( i = 0, 1, \ldots \text{do} \) |
| 3. \( t \leftarrow \lfloor p_{2i} \cdot |\mathcal{X}| \rfloor + 1; \) |
| 4. Let \( v \) be the \( t \)-th element of \( \mathcal{X} \); |
| 5. if \( p(2i + 1) \leq x_v \) then |
| 6. \textbf{return} \( v \); |
is sampled, but it is only used for sampling $\hat{w}$. Therefore, it is permissible to replace the sampling of $x(e)$ with the operation of sampling an integer value $\hat{w}(e)$. This can be rewritten using UniformDiscrete as follows. If $\hat{w}(e)$ is greater than $12\epsilon^{-1}|V| + 3$, then no path corresponding to $e$ is created in $G$, so if $l(e) > 12\epsilon^{-1}|V| + 1$, do nothing. Otherwise, let $X = \{2, \ldots, [12\epsilon^{-1}|V| + 4]\}$. Let a probability vector $y$ be

$$y(i) = \begin{cases} \frac{l(e) + 1 - w(e)}{b} & (i = l(e) + 2) \\ 1 - \frac{l(e) + 1 - w(e)}{b} & (i = l(e) + 3) \\ 0 & \text{otherwise} \end{cases}$$

Then, $\hat{w}(e)$ is sampled by UniformDiscrete($X$, $y$, $p$).

We also need to make DiSP($\hat{G}$) compatible with the shared randomness setting. In Algorithm 2, we use UniformFixedRatio to sample $\gamma$ at Line 10 and to sample $d$ at Line 2, and use Uniform to sample $l$ at Line 3. This makes Lemma III.16 and Lemma III.6 have their right-hand sides tripled, but it does not affect the analysis.

To sample $v$ in Line 7, we use UniformDiscrete. We would like to take the universe $X$ as the set of vertices $\hat{V}$ of the graph $\hat{G}$. However, because LipSP($G$, $w$, $s$, $t$, $e$) in Algorithm 3 may call DiSP($\hat{G}$, $\cdot$) for different graphs $\hat{G}$ depending on the choice of weights $w$ and internal randomness, we cannot set $X = \hat{V}$. For this reason, we cannot obtain a low-sensitivity algorithm for the shortest path problem in the shared randomness setting just by modifying Algorithm 2 alone. Instead, let us recall that each edge of $G$ corresponds to two directed paths of length at most $12\epsilon^{-1}|V| + 3$. Thus, any vertex $v$ in $\hat{G}$ that does not correspond to a vertex in $G$ can be represented as a triple $(e, t, i)$, where $e$ is an edge in $G$ that corresponds to $v$, $t \in 0, 1$ indicates whether $v$ belongs to the first or second path in $G$, and $i \in \{1, \ldots, 12\epsilon^{-1}|V| + 3\}$ is the distance from the start of the path. Therefore, we can take $X = V \cup \{(E \times \{0, 1\} \times \{1, \ldots, 12\epsilon^{-1}|V| + 3\}\}$, and use UniformDiscrete to sample $v$.

3) Algorithm 4: To sample $b$ in Line 2 of Algorithm 4, we use Uniform([1, $\alpha$], $p$). In Line 3, we need to uniformly sample a permutation $\pi$ of $\{1, \ldots, |V|\}$. It can be achieved by repeatedly applying SampleDiscrete($V$, $U$, $|V| \setminus \{\pi(1), \ldots, \pi(k)\}$), $p$), where we abuse the notation $\hat{U}[X]$ to represent the distribution that samples elements in $X$ uniformly at random, and elements in $V \setminus X$ with probability 0. The analysis is not affected by these modifications, so the maximum weight matching problem can be approximated with a $(1/8 - \epsilon)$ approximation and a Lipschitz constant of $O(\epsilon^{-1})$ in the shared randomness setting.

4) Algorithm 5: To sample $b$ in Line 2 of Algorithm 5, we use UniformFixedRatio($\cdot$). This triples the right-hand side of Lemma VII.4, but it does not affect the analysis. Moreover, Line 4 samples $\hat{w}(e)$ using UniformFixedWidth($w(e)$, $w(e) + b$, $p$). This also does not affect the analysis, so in the shared randomness setting, a $(1 - \epsilon)$ approximation algorithm with pointwise Lipschitzness $O(\epsilon^{-1}n/\text{opt})$ can be obtained for the minimum weight spanning tree problem.

5) Algorithm 6: To sample $B$ in Line 2 of Algorithm 6, UniformFixedRatio($\cdot$). This triples the right-hand side of Lemma VII.16 but does not affect the analysis. We use SampleDiscrete($V$ \cup \{\}, ($x_{i,1}, \ldots, x_{i,|V|}, 1 - \sum_{j=1}^{V} x_{i,j}$), $p$) to sample $p(i)$ in Line 6, and SampleDiscrete($U$, $U$($C_j$), $p$) to sample $q(j)$ in Line 11. These do not affect the analysis of Lemma VII.15, and thus we obtain a $(1/2 - \epsilon)$ approximation algorithm with pointwise Lipschitzness $O\left(\frac{\epsilon^{-1}|E|}{\text{opt}(G, w)}\right)$ for the maximum weight bipartite matching problem even in the shared randomness setting.

Acknowledgments

S.K. is supported by JST, PRESTO Grant Number JPMJPR192B and JSPS KAKENHI Grant Number JP21J20547. Y.Y. is supported by JST, PRESTO Grant Number JPMJPR192B and JSPS KAKENHI Grant Number JP20H05965.

References

[1] Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. IEEE Access, 6:14410–14430, 2018.

[2] Sayan Bhattacharya, Fabrizio Grandoni, and David Wajc. Online edge-coloring algorithms via the nibble method. In Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA), pages 2830–2842. SIAM, 2021.

[3] Sayan Bhattacharya, Silvio Lattanzi, and Nikos Parotsidis. Efficient and stable fully dynamic facility location. In Advances in Neural Information Processing Systems, 2022.

[4] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In 2017 IEEE Symposium on Security and Privacy (SP), pages 39–57. Ieee, 2017.

[5] Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In International Conference on Machine Learning, pages 854–863. PMLR, 2017.

[6] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. Advances in neural information processing systems, 26, 2013.

[7] Hossein Esfandiari, Alkis Kalavasis, Amin Karbasi, Andreas Krause, Vahab Mirrokni, and Grigoris Velegkas. Replicable bandits. In The 11th International Conference on Learning Representations (ICLR), 2023. URL: https://openreview.net/forum?id=gcD2UtCGMc2.
Proof of Lemma 1.7. First, note that the inequality in the statement holds even for $\delta > c$ because then
\[
\text{EM}_w((A(G, w), w), (A(G, w + \delta 1_f), w + \delta 1_f)) 
\leq \text{EM}_w((A(G, w), w), (A(G, w + (\delta - c)1_f), w + (\delta - c)1_f)) 
\quad + \text{EM}_w((A(G, w + (\delta - c)1_f), w) + (\delta - c)1_f), 
\quad (A(G, w + \delta 1_f), w + \delta 1_f)) 
\quad \leq (\delta - c)L + cL = \delta L,
\]
where the last inequality can be proven by induction. Also because EM$_w$ is commutative, the inequality
\[
\text{EM}_w((A(G, w), w), (A(G, w'), w')) 
\leq \sum_{i=1}^m \text{EM}_w((A(G, w_{i-1}), w_{i-1}), (A(G, w_i), w_i))
\]
holds even for $\delta < 0$.

Let $e_1, \ldots, e_m$ be an arbitrary ordering of elements in $E$, where $m = |E|$. For two weight vectors $w, w' \in \mathbb{R}^E_{\geq 0}$, by the triangle inequality, we have
\[
\text{EM}_w((A(G, w), w), (A(G, w'), w')) 
\leq \sum_{i=1}^m \text{EM}_w((A(G, w_{i-1}), w_{i-1}), (A(G, w_i), w_i)),
\]
where $w_i \in \mathbb{R}^E_{\geq 0}$ is the weight vector given by
\[
w_i(e_j) = \begin{cases} 
  w'(e_j) & (j \leq i) \\
  w(e_j) & (j > i). 
\end{cases}
\]
Since $w_i = w_{i-1} + (w'(e_i) - w(e_i)) 1_{e_i}$, we have
\[
\sup_{w, w' \in \mathbb{R}^E_{\geq 0}} \text{EM}_w((A(G, w), w), (A(G, w'), w')) 
\leq \sup_{w, w' \in \mathbb{R}^E_{\geq 0}} \frac{\sum_{i=1}^m \text{EM}_w((A(G, w_{i-1}), w_{i-1}), (A(G, w_i), w_i))}{\sum_{i=1}^m |w'(e_i) - w(e_i)|}
\leq \sup_{w, w' \in \mathbb{R}^E_{\geq 0}} \max \sum_{i=1}^m |w'(e_i) - w(e_i)|.
\]

APPENDIX
Proof of Lemma VII.1. Let \( e_1, \ldots, e_m \) be an arbitrary ordering of elements in \( E \), where \( m = |E| \). For two weight vectors \( w, w' \in \mathbb{R}_E \geq 0 \) with \( \|w - w'\|_1 \leq c \), by the triangle inequality, we have

\[
EM(u) (A(G, w), A(G, w')) \leq \sum_{i=1}^{m} EM(u) (A(G, w_{i-1}), A(G, w_i)),
\]

where \( w_i \in \mathbb{R}_E \) is defined as in the proof of Lemma I.7. Since \( w_i = w_{i-1} + (w'(e_i) - w(e_i))1_{e_i} \), we have

\[
\sup_{w, w' \in \mathbb{R}_E \geq 0, \ w \neq w'} \frac{EM(u) (A(G, w), A(G, w'))}{\|w - w'\|_1} \leq \sup_{w, w' \in \mathbb{R}_E \geq 0, \ w \neq w'} \frac{\sum_{i=1}^{m} EM(u) (A(G, w_{i-1}), A(G, w_i))}{\sum_{i=1}^{m} |w'(e_i) - w(e_i)|} \leq \sup_{w, w' \in \mathbb{R}_E \geq 0, \ w \neq w'} \max_{i=1, \ldots, m} \frac{EM(u) (A(G, w_{i-1}), A(G, w_i))}{|w'(e_i) - w(e_i)|} \leq L,
\]

where the last inequality is from the fact that \( \|w - w_{i-1}\|_1 \leq \|w - w'\|_1 \leq c \) for \( w(e_i) \leq w'(e_i) \) and \( \|w - w_i\|_1 \leq \|w - w'\|_1 \leq c \) for \( w(e_i) > w'(e_i) \). Therefore, \( A \) has pointwise Lipschitz constant \( L \) on \( G \) at \( w \), considering the neighborhood \( \{w': \|w - w'\| \leq c\} \). \( \square \)