Average Sensitivity of Graph Algorithms

Nithin Varma  
University of Haifa  
nvarma@bu.edu

Yuichi Yoshida  
National Institute of Informatics  
yyoshida@nii.ac.jp

Abstract

In modern applications of graph algorithms, where the graphs of interest are large and dynamic, it is unrealistic to assume that an input representation contains the full information of a graph being studied. Hence, it is desirable to use algorithms that, even when provided with only a (large) subgraph, output solutions that are close to the solutions output when the whole graph is available. We formalize this feature by introducing the notion of average sensitivity of graph algorithms, which is the average earth mover's distance between the output distributions of an algorithm on a graph and its subgraph obtained by removing an edge, where the average is over the edges removed and the distance between two outputs is the Hamming distance.

In this work, we initiate a systematic study of average sensitivity. After deriving basic properties of average sensitivity such as composition, we provide efficient approximation algorithms with low average sensitivities for concrete graph problems, including the minimum spanning forest problem, the global minimum cut problem, the $s$-$t$ minimum cut problem, the balanced cut problem, and the maximum matching problem. In addition, we prove that the average sensitivity of our global minimum cut algorithm is almost optimal, by showing a nearly matching lower bound. We also show that every algorithm for the 2-coloring problem has average sensitivity linear in the number of vertices. One of the main ideas involved in designing our algorithms with low average sensitivity is an algorithm with low average sensitivity for solving linear programs with respect to a related notion of stability. We also establish and utilize the following fact: if the presence of a vertex or an edge in the solution output by an algorithm can be decided locally, then the algorithm has a low average sensitivity, allowing us to reuse the analyses of known sublinear-time algorithms.
1 Introduction

In modern applications of graphs algorithms, where the graphs of interest are large and dynamic, it is unrealistic to assume that an input representation contains the full information of a graph being studied. For example, consider a social network, where a vertex corresponds to a user of the social network service and an edge corresponds to a friendship relation. It is reasonable to assume that users do not always update new friendship relations on the social network service, and that sometimes they do not fully disclose their friendship relations because of security or privacy reasons. Hence, we can only obtain an approximation $G'$ to the true social network $G$. This brings out the need for algorithms that can extract information on $G$ by solving a problem on $G'$. Moreover, as the solutions output by a graph algorithm are often used in applications such as detecting communities [27,28], ranking nodes [31], and spreading influence [19], the solutions output by an algorithm on $G'$ should be close to those output on $G$.

We assume that the $n$-node input graph $G'$ at hand is a randomly chosen (large) subgraph of an unknown true graph $G$. Intuitively, a deterministic algorithm $A$ is stable-on-average when the Hamming distance $d_{\text{ham}}(A(G), A(G'))$ is small, where $A(G)$ and $A(G')$ are outputs of $A$ on $G$ and $G'$, respectively. Here, outputs are typically vertex sets or edges sets and we assume that they are represented appropriately using binary strings. More specifically, for an integer $k \geq 1$, we say that the $k$-average sensitivity of a deterministic algorithm $A$ is

$$\mathbb{E}_{\{e_1,\ldots,e_k\}\sim\binom{E}{k}}\left[d_{\text{ham}}(A(G), A(G - \{e_1,\ldots,e_k\}))\right]$$

for every graph $G = (V, E)$, where $G - F$ for an edge set $F$ is the subgraph obtained from $G$ by removing $F$, where $\{e_1,\ldots,e_k\}$ is sampled uniformly at random from $\binom{E}{k}$, the set of all subsets of $E$ of cardinality $k$. When $k = 1$, we call the $k$-average sensitivity simply average sensitivity. We say that algorithms with low $k$-average sensitivity are $k$-stable-on-average. Although we focus on graphs here, we note that our definition can also be extended to the study of combinatorial objects other than graphs such as strings and constraint satisfaction problems. Since average sensitivity does not care about the solution quality, an algorithm that outputs the same solution regardless of the input has the least possible average sensitivity, though it is definitely useless. Hence, the key question in a study of average sensitivity is to reveal the trade-off between solution quality and average sensitivity for various problems.

Example 1.1. Consider the algorithm that, given a graph $G = (V, E)$ on $n$ vertices, outputs the set of vertices of degree at least $n/2$. As removing an edge changes the degree of exactly two vertices, the sensitivity of this algorithm is at most 2.

Example 1.2. Consider the $s$-$t$ shortest path problem, where given a graph $G = (V, E)$ and two vertices $s, t \in V$, we are to output the set of edges in a shortest path from $s$ to $t$. Since the length of a shortest path is always bounded by $n$, where $n$ is the number of vertices, every deterministic algorithm has average sensitivity $O(n)$. Indeed, there exists a graph for which this trivial upper bound is tight. Think of a cycle of even length $n$ and two vertices $s, t$ in diametrically opposite positions. Consider an arbitrary deterministic algorithm $A$, and assume that it outputs a path $P$ (of length $n/2$) among the two shortest paths from $s$ to $t$. With probability half, an edge in $P$ is removed, and $A$ must output the other path $Q$ (of length $n/2$) from $s$ to $t$. Hence, the average sensitivity must be $1/2 \cdot (n/2) = \Omega(n)$. In this sense, there is no deterministic algorithm with nontrivial average sensitivity for the $s$-$t$ shortest path problem.
We also generalize our definition of average sensitivity to apply to randomized algorithms. Let $\mathcal{A}(G)$ denote the output distribution of $\mathcal{A}$ on $G$. Let $d_{EM}(\mathcal{A}(G), \mathcal{A}(G'))$ denote the earth mover’s distance between $\mathcal{A}(G)$ and $\mathcal{A}(G')$, where the distance between two outputs is measured by the Hamming distance. Then, for an integer $k \geq 1$, the $k$-average sensitivity of a randomized algorithm $\mathcal{A}$ is
\[
\mathbb{E}_{\{e_1, \ldots, e_k\} \sim \binom{E}{k}} \left[ d_{EM}(\mathcal{A}(G), \mathcal{A}(G - \{e_1, \ldots, e_k\})) \right]
\] (2)
where $\{e_1, \ldots, e_k\}$ is sampled uniformly at random from $\binom{E}{k}$. Note that when the algorithm $\mathcal{A}$ is deterministic, (2) matches the definition of the average sensitivity for deterministic algorithms.

**Remark 1.3.** The $k$-average sensitivity of an algorithm $\mathcal{A}$ with respect to the total variation distance can be defined as $\mathbb{E}_{\{e_1, \ldots, e_k\} \sim \binom{E}{k}} \left[ d_{TV}(\mathcal{A}(G), \mathcal{A}(G - \{e_1, \ldots, e_k\})) \right]$, where $d_{TV}(\cdot, \cdot)$ denotes the total variation distance. It is easy to observe that, if the $k$-average sensitivity of an algorithm with respect to the total variation distance is at most $\gamma(G)$, then its $k$-average sensitivity is bounded by $H \cdot \gamma(G)$, where the $H$ is the maximum Hamming weight of a solution output by $\mathcal{A}$ on $G$.

**Example 1.4.** Randomness does not help improve the average sensitivity of algorithms for the $s$-$t$ shortest path problem. Think of the cycle graph given in Example 1.2 and suppose that a randomized algorithm $\mathcal{A}$ outputs $P$ and $Q$ with probability $p$ and $q = 1 - p$, respectively. Then, the average sensitivity is $p \cdot (1/2 \cdot (n/2)) + q \cdot (1/2 \cdot (n/2)) = \Omega(n)$.

### 1.1 Basic properties of average sensitivity

The definition of average sensitivity has many nice properties. In this section, we discuss some useful properties of average sensitivity that we use as building blocks in the design of our stable-on-average algorithms. We denote by $\mathcal{G}$ the (infinite) set consisting of all graphs. Given a graph $G = (V, E)$ and $e \in E$, we use $G - e$ as a shorthand for $G - \{e\}$. We use $n$ and $m$ to denote the number of vertices and edges in the input graph, respectively.

**Bounds on $k$-average sensitivity from bounds on average sensitivity.** This is one of the most important properties of our definition of average sensitivity. It essentially says that bounding the average sensitivity of an algorithm with respect to the removal of a single edge automatically gives a bound on the average sensitivity of that algorithm with respect to the removal of multiple edges. In other words, it is enough to analyze the average sensitivity of an algorithm with respect to the removal of a single edge.

**Theorem 1.5.** Let $\mathcal{A}$ be an algorithm for a graph problem with the average sensitivity given by $f(n,m)$. Then, for any integer $k \geq 1$, the algorithm $\mathcal{A}$ has $k$-average sensitivity at most $\sum_{i=1}^{k} f(n,m - i + 1)$.

In particular, if the average sensitivity is a nondecreasing function of the number of edges, the above theorem immediately implies that the $k$-average sensitivity is at most $k$ times the average sensitivity.
Sequential composition. It will be useful if we can obtain a stable-on-average algorithm by sequentially applying several stable-on-average subroutines. We show two different sequential composition theorems for average sensitivity.

**Theorem 1.6** (Sequential composition). Consider two randomized algorithms $A_1: G \rightarrow S_1, A_2: G \times S_1 \rightarrow S_2$. Suppose that the average sensitivity of $A_1$ with respect to the total variation distance is $\gamma_1$ and the average sensitivity of $A_2(\cdot, S_1)$ is $\beta_2^{(S_1)}$ for any $S_1 \in S_1$. Let $A: G \rightarrow S_2$ be a randomized algorithm obtained by composing $A_1$ and $A_2$, that is, $A(G) = A_2(G, A_1(G))$. Then, the average sensitivity of $A$ is $H \cdot \gamma_1(G) + E_{S_1 \sim A_1(G)} \left[ \beta_2^{(S_1)}(G) \right]$, where $H$ denotes the maximum Hamming weight among those of solutions obtained by running $A$ on $G$ and $\{G - e\}$ over all $e \in E$.

Our second composition theorem is for the average sensitivity with respect to the total variation distance. This is also useful for analyzing the average sensitivity with respect to the earth mover’s distance, as it can be bounded by the average sensitivity with respect to the total variation distance times the maximum Hamming weight of a solution, as in Remark [1.3].

**Theorem 1.7** (Sequential composition w.r.t. the TV distance). Consider $k$ randomized algorithms $A_i: G \times \prod_{j=1}^{i-1} S_j \rightarrow S_i$ for $i \in \{1, \ldots, k\}$. Suppose that, for each $i \in \{1, \ldots, k\}$, the average sensitivity of $A_i(\cdot, S_1, \ldots, S_{i-1})$ is $\gamma_i$ with respect to the total variation distance for every $S_1 \in S_1, \ldots, S_{i-1} \in S_{i-1}$. Consider a sequence of computations $S_1 = A_1(G), S_2 = A_2(G, S_1), \ldots, S_k = A_k(G, S_1, \ldots, S_{k-1})$. Let $A: G \rightarrow S_k$ be a randomized algorithm that performs this sequence of computations on input $G$ and outputs $S_k$. Then, the average sensitivity of $A$ with respect to the total variation distance is at most $\sum_{i=1}^{k} \gamma_i(G)$.

Parallel composition. It is often the case that there are multiple algorithms that solve the same problem albeit with different average sensitivity guarantees. Such stable-on-average algorithms can be composed by running them according to a distribution determined by the input graph. The advantage of such a composition, which we call a parallel composition, is that the average sensitivity of the resulting algorithm might be better than that of the component algorithms.

**Theorem 1.8** (Parallel composition). Let $A_1, A_2, \ldots, A_k$ be algorithms for a graph problem with average sensitivities $\beta_1, \beta_2, \ldots, \beta_k$, respectively. Let $A$ be an algorithm that, given a graph $G$, runs $A_i$ with probability $\rho_i(G)$ for $i \in [k]$, where $\sum_{i \in [k]} \rho_i(G) = 1$. Let $H$ denote the maximum Hamming weight among those of solutions obtained by running $A$ on $G$ and $\{G - e\}_{e \in E}$. Then the average sensitivity of $A$ is at most $\sum_{i \in [k]} \rho_i(G) \cdot \beta_i(G) + H \cdot E_{e \sim E} \left[ \sum_{i \in [k]} |\rho_i(G) - \rho_i(G - e)| \right]$.

In this paper, we use the above theorem extensively to combine algorithms with different average sensitivities.

### 1.2 Connection to sublinear-time algorithms

We show a relationship between the average sensitivity of an algorithm and the query complexity of a local algorithm [29], [33], [36] that simulates oracle access to the solution output by the former algorithm. Roughly speaking, we show, in Theorem [1.3], that the average sensitivity of an algorithm $A$ is bounded by the query complexity of another algorithm $O$, where $O$ makes queries to a graph $G$ and simulates oracle access to the solution produced by $A$ on input $G$. We use Theorem [1.9] to prove the existence of stable-on-average matching algorithms based on the sublinear-time matching algorithms due to [36].
Theorem 1.9 (Locality implies low average sensitivity). Consider a randomized algorithm \( A : G \to S \) for a graph problem, where each solution output by \( A \) is a subset of the set of edges of the input graph. Assume that there exists an oracle \( O \) satisfying the following:

- when given access to a graph \( G = (V, E) \) and query \( e \in E \), the oracle generates a random string \( \pi \in \{0, 1\}^{|V|} \) and outputs whether \( e \) is contained in the solution obtained by running \( A \) on \( G \) with \( \pi \) as its random string,

- the oracle \( O \) makes at most \( q(G) \) queries to \( G \) in expectation, where this expectation is taken over the random coins of \( A \) and a uniformly random query \( e \in E \).

Then, \( A \) has average sensitivity at most \( q(G) \). Moreover, given the promise that the input graphs satisfy \( |E| \geq |V| \), the statement applies also to algorithms for which each solution is a subset of the vertex set of the input graph.

Theorem 1.9 cements the intuition that strong locality guarantees for solutions output by an algorithm imply that the removal of edges from a graph affects only the presence of a few edges in the solution, which in turn implies low average sensitivity. Due to its applicability in bounding the average sensitivity of algorithms, we think that Theorem 1.9 could lead to further research in the design of local algorithms for various graph problems.

1.3 Stable-on-average algorithms for concrete problems

We summarize, in Table 1, the average sensitivity bounds that we obtain for various concrete problems. All our algorithms run in polynomial time, and the bounds on \( k \)-average sensitivity of these algorithms can be easily derived using Theorem 1.5. Henceforth, let \( n, m, \text{OPT} \) denote the number of vertices, the number of edges, and the optimal value. To help interpret our bounds on average sensitivity, we mention that for maximization problems whose optimal values are sufficiently Lipschitz with respect to edge removals, \( O(\text{OPT}) \) is a trivial upper bound for the average sensitivity. However, this is not the case, in general, for minimization problems.

For the minimum spanning forest problem, we show that Kruskal’s algorithm [20] achieves average sensitivity \( O(n/m) \), which is quite small regarding that the spanning forest can have \( \Omega(n) \) edges. In contrast, it is not hard to show that the average sensitivities of the known polynomial time (approximation) algorithms for the other problems listed in Table 1 are all \( \Omega(n) \).

For the global minimum cut problem, our algorithm outputs a cut as a vertex set. As the approximation ratio of our algorithm is constant, it is likely to output a cut of size close to \( \text{OPT} \), and hence we want to make its average sensitivity smaller than \( \text{OPT} \). We observe that the average sensitivity becomes smaller than \( \text{OPT} \) when \( \text{OPT} = \Omega(t \log \log t \log t) \) for \( t = \log(n)/\epsilon \), and it quickly decreases as \( \text{OPT} \) increases.

We also prove a nearly tight lower bound on the average sensitivity of any algorithm that outputs a purely multiplicative approximation to the minimum cut size. In particular, when \( \text{OPT} \) is \( o(\log n) \), our lower bound matches, up to a polylogarithmic factor, our upper bound for 3-approximating the minimum cut size. When \( \text{OPT} = \Omega(\log n) \), our algorithm to 3-approximate the minimum cut size has average sensitivity \( O(1) \).

Our stable-on-average algorithms for both the minimum \( s-t \) cut and balanced cut problems output cuts as vertex sets. Both the algorithms have strong guarantees on the average sensitivity: specifically, their average sensitivities are polylogarithmic in \( n \).
Table 1: Our results. Here $n$, $m$, $\text{OPT}$ denote the number of vertices, the number of edges, and the optimal value, respectively, and $\epsilon \in (0, 1)$ is an arbitrary constant. The notation $\tilde{O}(\cdot)$ and $\tilde{\Omega}(\cdot)$ hides polylogarithmic factors in $n$ and $\log n$, respectively. An approximation guarantee of the form $(\alpha, \beta)$ indicates a multiplicative loss of $\alpha$ and an additive loss of $\beta$. Small additive losses in the approximation guarantees are omitted.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Output</th>
<th>Approximation Ratio</th>
<th>Average Sensitivity</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum Spanning Forest</td>
<td>Edge set</td>
<td>$1$</td>
<td>$O\left(\frac{n}{m}\right)$</td>
<td>Sec. 3</td>
</tr>
<tr>
<td>Global Minimum Cut</td>
<td>Vertex set</td>
<td>$2 + \epsilon$</td>
<td>$n^{O\left(\frac{1}{\OPT}\right)}$</td>
<td>Sec. 4.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$&lt; \infty$</td>
<td>$\Omega(n)$</td>
<td>Sec. 4.2</td>
</tr>
<tr>
<td>s-t Minimum Cut</td>
<td>Vertex set</td>
<td>$(1 + \epsilon, O(\log n))$</td>
<td>$O\left(\frac{\log n}{\epsilon}\right)$</td>
<td>Sec. 6</td>
</tr>
<tr>
<td>Balanced Cut</td>
<td>Vertex set</td>
<td>$(\tilde{O}\left(\frac{\log^4 n}{\epsilon}\right), \tilde{O}\left(\frac{\log^5 n}{\epsilon}\right))$</td>
<td>$\tilde{O}\left(\frac{\log^3 n}{\epsilon}\right)$</td>
<td>Sec. 7</td>
</tr>
<tr>
<td>Maximum Matching</td>
<td>Edge set</td>
<td>$1/2$</td>
<td>$1$</td>
<td>Sec. 8.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$1 - \epsilon$</td>
<td>$\tilde{O}\left(\frac{\text{OPT}}{\epsilon^3}\right)^{\frac{1}{1+\Omega(\epsilon^2)}}$</td>
<td>Sec. 8.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$1$</td>
<td>$\Omega(n)$</td>
<td>Sec. 8.4</td>
</tr>
<tr>
<td>Minimum Vertex Cover</td>
<td>Vertex set</td>
<td>$2$</td>
<td>$2$</td>
<td>Sec. 8.2</td>
</tr>
<tr>
<td>2-Coloring</td>
<td>Vertex set</td>
<td>—</td>
<td>$\Omega(n)$</td>
<td>Sec. 9</td>
</tr>
</tbody>
</table>

Our lower bound of $\Omega(n)$ on the average sensitivity of algorithms that output the exact global minimum cut also applies to the minimum s-t cut problem. In contrast, for every constant $\epsilon > 0$, we obtain an approximation algorithm for the minimum s-t cut problem with average sensitivity $O(\log(n)/\epsilon)$ that outputs a cut of size at most $(1 + \epsilon) \cdot \text{OPT} + O(\log n)$. Note that when $\text{OPT} = \Omega(\log n)$, our algorithm can be thought of as a 2-approximation algorithm for the minimum s-t cut problem.

Our stable-on-average algorithm for the balanced cut problem incur only polylogarithmic multiplicative and additive losses in $n$. The cut-sets corresponding to its output have cardinalities at least $\Omega\left(\frac{n}{\text{polylog} n}\right)$ ensuring that the cut is quite balanced. The previous best polynomial-time algorithm, whose average sensitivity is hard to bound, has approximation ratio $O(\sqrt{\log n})$ and outputs a vertex set of size $\Omega(n)$ $\text{[9]}$. These guarantees are only slightly better than those of our stable-on-average algorithm.

We show that the average sensitivity of every algorithm that outputs the exact maximum matching is $\Omega(n)$, implying that some approximation is essential to obtain nontrivial average sensitivity. We also propose two stable-on-average approximation algorithms for maximum matching. Our first algorithm has approximation ratio $1/2$ and average sensitivity $O(1)$. This result immediately implies a 2-approximation algorithm for the minimum vertex cover problem with constant average sensitivity. Our second algorithm for maximum matching has approximation ratio $1 - \epsilon$ and average
sensitivity $\tilde{O}\left((\text{OPT}/\varepsilon^3)^{1/(1+\Omega(\varepsilon^2))}\right)$ for every constant $\varepsilon \in (0, 1)$.

In the 2-coloring problem, given a bipartite graph, we are to output one part in the bipartition. For this problem, we show a lower bound of $\Omega(n)$ for the average sensitivity: that is, there is no algorithm with nontrivial average sensitivity.

1.4 Discussions on average sensitivity

Output representation. The notion of average sensitivity is dependent on the output representation. For example, we can double the average sensitivity by duplicating the output. A natural idea for alleviating this issue is to normalize the average sensitivity by the maximum Hamming weight $H$ of a solution. However, for minimization problems where the optimal value $\text{OPT}$ could be much smaller than $H$, such a normalization can diminish subtle differences in average sensitivity, e.g., $O(\text{OPT}^{1/2})$ vs $O(\text{OPT})$. It is an interesting open question whether there is a canonical way to normalize average sensitivity so that the resulting quantity is independent of the output representation.

Sensitivity against adversarial edge removals. It is also natural to take the maximum, instead of the average, over edges in definitions (1) and (2), which can be seen as sensitivity against adversarial edge removals. Indeed a similar notion has been proposed to study algorithms for geometric problems [25]. However, in the case of graph algorithms, it is hard to guarantee that the output of an algorithm does not change much after removing an arbitrary edge. Moreover, by a standard averaging argument, one can say that for 99% of arbitrary edge removals, the sensitivity of an algorithm is asymptotically equal to the average sensitivity, which is sufficient in most cases.

Average sensitivity w.r.t. edge additions. As another variant of average sensitivity, it is natural to consider incorporating edge additions in definitions (1) and (2). If an algorithm is stable against edge additions, then in addition to the case of not knowing the true graph as we have discussed earlier, it will be useful for the case that the graph dynamically changes but we want to prevent the output of the algorithm from fluctuating too much. However, in contrast to removing edges, it is not always clear how we should add edges to the graph in definitions (1) and (2). A naive idea is sampling $k$ pairs of vertices uniformly at random and adding edges between them. This procedure makes the graph close to a graph sampled from the Erdős-Rényi model [9], which does not represent real networks such as social networks and road networks well. To avoid this subtle issue, in this work, we focus on removing edges.

Alternative notion of average sensitivity for randomized algorithms. Consider a randomized algorithm $A$ that, given a graph $G$ on $n$ vertices, generates a random string $\pi \in \{0, 1\}^{r(n)}$ for some function $r : \mathbb{N} \to \mathbb{N}$, and then runs a deterministic algorithm $A_\pi$ on $G$, where the algorithm $A_\pi$ has $\pi$ hardwired into it. Assume that $A_\pi$ can be applied to any graph. It is also natural to define the average sensitivity of $A$ as

$$E_{\pi \sim \text{Ham}} \left[ E_{e \sim E} \left[ d_{\text{Ham}}(A_\pi(G), A_\pi(G - e)) \right] \right].$$

In other words, we measure the expected distance between the outputs of $A$ on $G$ and $G - e$ when we feed the same string $\pi$ to $A$, over the choice of $\pi$ and edge $e$. Note that (3) upper bounds (2).
because, in the definition of the earth mover's distance, we optimally transport probability mass from $A(G)$ to $A(G - e)$ whereas, in (3), how the probability mass is transported is not necessarily optimal.

We can actually bound (3) for some of our algorithms. In this work, however, we focus on the definition (2) because the assumption that $A_{\pi}$ can be applied to any graph does not hold in general, and bounding (3) is unnecessarily tedious and is not very enlightening.

1.5 Overview of our techniques

Minimum spanning forest. For the minimum spanning forest problem, we show that the classical Kruskal’s algorithm has low average sensitivity; specifically, at most 1. Interestingly, Kruskal’s algorithm is deterministic and yet has low average sensitivity. In contrast, we show that Prim’s algorithm can have average sensitivity $\Omega(m)$ for a natural (and deterministic) rule of breaking ties among edges.

Global minimum cut. For the global minimum cut problem, our algorithm is inspired by a differentially private algorithm by Gupta et al. [13]. Our algorithm, given a parameter $\varepsilon > 0$ and a graph $G$ as input, first enumerates a list of cuts whose sizes are at most $(2 + \varepsilon) \cdot \text{OPT}$; this enumeration can be done in polynomial time as shown by Karger’s theorem [15]. It then outputs a cut from the list with probability exponentially small in the product of the size of the cut and $O(1/\varepsilon \cdot \text{OPT})$. The main argument in analyzing the average sensitivity of the algorithm is that the aforementioned distribution is very close (in earth mover’s distance) to a related Gibbs distribution on the set of all cuts in the graph. Therefore the average sensitivity of the algorithm is of the same order as that of the average sensitivity of sampling a cut from such a Gibbs distribution doing which requires exponential time. We finally show that the average sensitivity of sampling a cut from this Gibbs distribution is at most $n^{O(1/\varepsilon \cdot \text{OPT})}$.

s-t minimum cut. The main tool used in obtaining our stable-on-average algorithm for the s-t minimum cut problem is a stable-on-average algorithm to solve linear programming relaxations with respect to a related notion of stability. Specifically, given a polytope $K \subseteq [0,1]^n$ (defined by the constraints of a linear program) and an objective function vector $c \in \{0,1\}^n$, the average sensitivity of an algorithm $\mathcal{A}$ to approximate an optimizer of the linear program (LP) is defined to be

$$\mathbb{E}_{i \in [n]} \left[ d_{\text{EM}}^i (\mathcal{A}(c), \mathcal{A}(c^{-i})) \right],$$

where $c^{-i}$ denotes the vector $c$ with the $i$-th entry set to 0, and $d_{\text{EM}}^i$ denotes the earth mover’s distance between distributions with respect to the $\ell_1$ distance. We first show that for every constant $\eta > 0$, the simple method of sampling a vector $x \in K$ with probability proportional to $e^{-\eta \langle c, x \rangle}$ has average sensitivity $O(\eta \text{OPT} + \log n)$ and outputs a vector with the expected cost of at most $\text{OPT} + \frac{\log n}{\eta}$, where $\langle x, x' \rangle$ for vectors $x, x' \in [0,1]^n$ denotes the dot product of $x$ and $x'$, and $\text{OPT} = \min_{x \in K} \langle c, x \rangle$. We then use a result of Lovasz and Vempala [22] to show that it is indeed possible to sample from a distribution close to the aforementioned (Gibbs) distribution in polynomial time. We further improve the average sensitivity of this procedure by first sampling the parameter $\eta$ from the Laplace distribution tightly concentrated around $\frac{\log n}{\varepsilon \cdot \text{OPT}}$ for a parameter $\varepsilon$ and then using the
sampled \( \eta \) in the above algorithm. We show that the resulting procedure has average sensitivity \( O\left(\frac{\log n}{\epsilon} \right) \) and outputs a vector with the expected cost of at most \((1 + \epsilon)\text{OPT} + O(\log n)\).

We use our stable-on-average LP solver with logarithmic average sensitivity to solve a natural linear programming relaxation of the \( s-t \) mincut problem. Given a graph \( G = (V, E) \), the relaxation contains variables \( d(\{u, v\}) \in [0, 1] \) for each \( \{u, v\} \in \binom{V}{2} \), where these variables can be thought of as representing a pseudometric over the vertices. We keep \( d(\{s, t\}) = 1 \) as a constraint to our linear program, in addition to triangle inequality constraints. Intuitively, if \( d(\{u, v\}) \) is large in a solution to the linear program, then the vertices \( u \) and \( v \) fall on different sides of the \( s-t \) cut represented by the solution. Given a solution to the relaxation, our rounding procedure samples a threshold \( \tau \in [0, 1] \) uniformly at random and outputs the set \( S \) consisting of all vertices \( u \in V \) such that \( d(\{s, u\}) \leq \tau \). The approximation guarantee of this algorithm follows from the fact that we are rounding based on a near optimal solution to the linear programming relaxation. To analyze the average sensitivity of the algorithm, we first show that the earth mover’s distance (with respect to Hamming distance) between the outputs of the rounding procedure for inputs \( d, d' \in [0, 1]^{\binom{V}{2}} \) is bounded by the \( \ell_1 \) distance between \( d \) and \( d' \). Combining this with the bound on the average sensitivity of our stable-on-average LP solver (with respect to the \( \ell_1 \) distance), we obtain our final bound on the average sensitivity for our algorithm to approximate the minimum \( s-t \) cut.

**Balanced cut.** Our stable-on-average algorithm for the balanced cut problem is based on the well known approximation algorithm for the sparsest cut problem due to Leighton and Rao [21]. Given a graph \( G = (V, E) \), the starting point of our algorithm is an LP relaxation that is quite similar to the one used in our algorithm for the \( s-t \) minimum cut. The only difference is that the special constraint \( d(\{s, t\}) = 1 \) is replaced with a constraint that the sum of the distance variables over all pairs of vertices is quadratic in \( n \). Such a constraint forces the sets on either side of the cut to have roughly the same size.

Given the aforementioned LP, we first use our stable-on-average LP solver to obtain a pseudometric \( d \in [0, 1]^{\binom{V}{2}} \), where we interpret \( d \) to represent distances between vertices. We then apply Bourgain’s embedding [4] to \( d \) and obtain a function \( f : V \mapsto \mathbb{R}^{k} \) such that for all \( u, v \in V \), the value \( d(\{u, v\}) \) is within a factor of \( O(\log n) \) of \( \|f(u) - f(v)\|_1 \), where \( k = O(\log^2 n) \). One can think of the Bourgain’s embedding as a low dimensional labeling of the vertices such that the distance between the labels of two vertices is roughly equal to the distance between them as given by the pseudometric.

In order to round the embedding and obtain the cut-set, we further reduce the dimension of the labeling by sampling (from an appropriate Gibbs distribution) an index \( j \in [k] \) such that \( \sum_{\{u,v\} \in \binom{V}{2}} |f_j(u) - f_j(v)| \) is within a polylogarithmic factor of a quadratic function of \( n \) (to ensure that the resulting cut is nearly balanced), where \( f_j : V \mapsto \mathbb{R} \) denotes the embedding \( f \) restricted to the \( j \)-th dimension. We then sample a threshold \( \tau \) uniformly at random from a small interval centered around the mean \( \frac{1}{n} \sum_{u \in V} f_j(u) \) and finally output the set of vertices \( u \) with \( f_j(u) < \tau \).

The fact that \( f \) is a low distortion embedding of \( d \) ensures that the size of the final cut output is not much different from the cut represented by the LP solution. The cut is nearly balanced because we sample the index \( j \in [k] \) from a distribution that favors indices with \( \text{large} \sum_{\{u,v\} \in \binom{V}{2}} |f_j(u) - f_j(v)| \), where this quantity is roughly equal (up to polylogarithmic factors) to \( \sum_{\{u,v\} \in \binom{V}{2}} d(\{u, v\}) \).

In order to show that the algorithm is stable-on-average, we first bound the average sensitivity of Bourgain’s embedding with respect to the \( \ell_1 \) distance. The average sensitivity of the rounding
procedure is obtained by combining (using a composition theorem) the bounds on average sensitivity of sampling the index $j$ according to the Gibbs distribution and the average sensitivity of sampling the threshold (both of which are stable-on-average). We then combine these two bounds with the average sensitivity bound of the LP solver in a straightforward manner and obtain our final bound on average sensitivity.

**Maximum matching.** Our stable-on-average $\frac{1}{2}$-approximation algorithm for the maximum matching problem considers a uniformly random ordering of the edges of the input graph and adds edges to the matching greedily according to that ordering. In the context of dynamic distributed algorithms, Censor-Hillel et al. [6] showed that at most 1 edge changes in the matching (in expectation) due to the removal of a uniformly random edge, where the expectation is taken over the edge removed and the ordering of edges. This result immediately implies that the average sensitivity of this randomized greedy matching algorithm is at most 1. In addition, it implies a 2-approximation algorithm for the minimum vertex cover problem with average sensitivity at most 2.

There are several components to the design and analysis of our stable-on-average $(1 − \varepsilon)$-approximation algorithm. Our starting point is the observation (Theorem 1.9) that the ability to locally simulate access to the solution of an algorithm $A$ implies that $A$ is stable-on-average. We use Theorem 1.9 to bound the average sensitivity of a $(1 − \varepsilon')$-approximation algorithm $A$ for the maximum matching problem, where $\varepsilon' = \Omega(\varepsilon)$. Specifically, $A$ constructs a matching by considering augmenting paths of increasing length and augmenting the matching (initially empty) iteratively, where the paths of each length are considered in a uniformly random order. Yoshida et al. [36] constructed a local algorithm that, given a uniformly random edge $e \in E$ as input, makes $O\left(\Delta O^{O(1/(\varepsilon')^2)}\right)$ queries to $G$ in expectation and answers whether $e$ is in the matching output by $A$ on $G$, where the expectation is over the choice of input $e$ and the randomness in $A$, and $\Delta$ is the maximum degree of $G$. Combined with Theorem 1.9, this implies that the average sensitivity of $A$ is $O\left(\Delta O^{O(1/(\varepsilon')^2)}\right)$.

Next, we transform $A$ to also work for graphs of unbounded degree as follows. The idea is to remove vertices of degree at least $\frac{m}{\varepsilon' \text{OPT}}$ from the graph and run $A$ on the resulting graph. This transformation affects the approximation guarantee only by an additive $\varepsilon' \text{OPT}$ term as the number of such high degree vertices is small. However, this thresholding procedure could in itself have high average sensitivity, since the thresholds for $G$ and $G - e$ can be very different for every $e \in E$.

We circumvent this issue by using a Laplace random variable $L$ as the threshold, where the distribution of $L$ is tightly concentrated around $\frac{m}{\varepsilon' \text{OPT}}$. We use our sequential composition theorem (Theorem 1.6) in order to analyze the average sensitivity of the resulting procedure, where we consider the instantiation of the Laplace random threshold as the first algorithm and the remaining steps in the procedure as the second algorithm. The first term in the expression given by Theorem 1.6 turns out to be a negligible quantity and is easy to bound. The main task in bounding the second term is to bound, for all $x \in \mathbb{R}$, the average sensitivity of a procedure $A_x$ that, on the input graph $G$, removes all vertices of degree at least $x$ from $G$ and runs the randomized greedy maximal matching algorithm. The heart of the argument in bounding this average sensitivity is that given a local algorithm $O$ with query complexity $q(\Delta)$ that simulates oracle access to the solutions output by an algorithm $A_x$, we can, for all $x \in \mathbb{R}$, construct a local algorithm $O_x$ for the algorithm $A_x$. Moreover, the query complexity of $O_x$ is at most $O(x^2q(x))$. By Theorem 1.9, this is also a bound on the average sensitivity of $A_x$. Using this, we bound the second term in the expression given by Theorem 1.6 as $\mathbb{E}_L [O(L^2q(L))] = O\left(\left(\frac{m}{\varepsilon' \text{OPT}}\right)^{O(1/(\varepsilon')^2)}\right)$. 

9
An issue with the aforementioned matching algorithm is that its average sensitivity is poor for graphs with small values of $OPT$. We observe that, in contrast to this, the algorithm that simply outputs the lexicographically smallest maximum matching has average sensitivity $O(\frac{OPT^2}{m})$, since the output matching stays the same unless an edge in the matching is removed. We obtain our final stable-on-average $(1 - \varepsilon)$-approximation algorithm for the maximum matching problem by running these two algorithms according to a probability distribution determined by the input graph. Using our parallel composition theorem, we bound the average sensitivity of the resultant algorithm as $\tilde{O}\left(\left(\frac{OPT}{\varepsilon^3}\right)^{1/(1+\Omega(\varepsilon^2))}\right)$.

2-coloring. To show our $\Omega(n)$ lower bound on the average sensitivity for 2-coloring, consider the set of all paths on $n$ vertices and the set of all graphs obtained by removing exactly one edge from these paths (called 2-part-paths). A path has exactly two ways of being 2-colored and a 2-path has four ways of being 2-colored. A path and 2-part-path are neighbors if the latter is obtained from the former by removing an edge. A 2-part-path has at most four neighbors. The output distribution of any 2-coloring algorithm $A$ on a 2-part-path can be close (in earth mover’s distance) only to those of at most 2 of its neighboring paths. If $A$, however, has low average sensitivity, the output distributions of $A$ have to be close on a large fraction of pairs of neighboring graphs, which gives a contradiction.

1.6 Related work

Average sensitivity of network centralities. (Network) centrality is a collective name for indicators that measure importance of vertices or edges in a network. Notable examples are closeness centrality [2, 3, 34], harmonic centrality [23], betweenness centrality [10], and PageRank [31]. To compare these centralities qualitatively, Murai and Yoshida [26] recently introduced the notion of average-case sensitivity for centralities. Fix a vertex centrality measure $c$; let $c_G(v)$ denote the centrality of a vertex $v \in V$ in a graph $G = (V,E)$. Then, the average-case sensitivity of $c$ on $G$ is defined as

$$S_c(G) = \mathbb{E}_{e \sim E} \mathbb{E}_{v \sim V} \frac{|c_{G-e}(v) - c_G(v)|}{c_G(v)},$$

where $e$ and $v$ are sampled uniformly at random. They showed various upper and lower bounds for centralities. See [26] for details.

Since a centrality measure assigns real values to vertices, they studied the relative change of the centrality values upon removal of random edges. As our focus in this work is on graph algorithms, our notion (2) measures the Hamming distance between solutions when one removes random edges.

Differential privacy. Differential privacy [7] is a notion closely related to average sensitivity. Assuming the existence of a neighbor relation over inputs, the definition of differential privacy requires that the distributions of outputs on neighboring inputs are similar. The variant of differential privacy closest to our definition of average sensitivity is edge differential privacy introduced by Nissim et al. [30] and further studied by [14, 13, 17, 18, 16, 32]. Here, the neighbors of a graph $G = (V,E)$ are defined to be $\{G - e\}_{e \in E}$. For $\varepsilon > 0$, we say that an algorithm is $\varepsilon$-differentially private if for all $e \in E$,

$$\exp(-\varepsilon) \cdot \Pr[A(G - e) \in S] \leq \Pr[A(G) \in S] \leq \exp(\varepsilon) \cdot \Pr[A(G - e) \in S]$$

(4)
for any set of solutions $S$.

Differential privacy has stricter requirements than average sensitivity. As differential privacy imposes the constraint \[(4)\] for every $e \in E$, the requirement is sometimes too strong for graph problems. For example, an algorithm that outputs a vertex cover of the input graph of size smaller than $n - 1$ is not differentially private. For example, for the minimum vertex cover problem, \[(4)\] implies that we must output a vertex cover for $G$ even for $G - e$, and it follows that we can only output a vertex cover of size at least $n - 1$. This is because the output reveals that there is no edge between two vertices that are not part of the vertex cover. It follows that we can only output a vertex cover of size at least $n - 1$. To avoid this issue, \[13\] considered outputting an implicit representation of a vertex cover.

Moreover, since differential privacy guarantees that the probabilities of outputting a specific solution on $G$ and $G - e$ are close to each other, the total variation distance between the two distributions $A(G)$ and $A(G - e)$ must be small. Since the earth mover’s distance between two output distributions can be small even if the total variation distance between them is large, even if an algorithm does not satisfy the conditions of differential privacy, it could still have small average sensitivity. Despite these differences, our stable-on-average algorithm for the global minimum cut problem is inspired by a differentially private algorithm for the same problem \[13\].

Generalization and stability of learning algorithms. Generalization \[35\] is a fundamental concept in statistical learning theory. Given samples $z_1, \ldots, z_n$ from an unknown true distribution $D$ over a dataset, the goal of a learning algorithm $L$ is to output a parameter $\theta$ that minimizes expected loss $\mathbb{E}_{z \sim D}[\ell(z; \theta)]$, where $\ell(z; \theta)$ is the loss incurred by a sample $z$ with respect to a parameter $\theta$. As the true distribution $D$ is unknown, a frequently used approach in learning is to compute a parameter $\theta$ that minimizes the empirical loss $\frac{1}{n} \cdot \sum_{i=1}^{n} \ell(z_i; \theta)$, which is an unbiased estimator of the expected loss and is purely a function of the available samples. The generalization error of a learner $L$ is a measure of how close the empirical loss is to the expected loss as a function of the sample size $n$.

One technique to reduce the generalization error is to add a regularization term to the loss function being minimized \[3\]. This also ensures that the learned parameter $\theta$ does not change much with respect to minor changes in the samples being used for learning. Therefore, in a sense, learning algorithms that use regularization can be considered as being stable according to our definition of sensitivity.

Bousquet and Elisseeff \[5\] defined a notion of stability for learning algorithms in relation to reducing the generalization error. Their stability notion requires that the empirical loss of the learning algorithm does not change much by removing or replacing any sample in the input data. In contrast, in our definition of average sensitivity, we consider removing random edges from a graph and measure the change in the output solution rather than that in the objective value.

1.7 Organization

We show our stable-on-average algorithms for the minimum spanning forest problem, the global minimum cut problem, the $s$-$t$ minimum cut problem, the balanced cut problem, and the maximum matching problem problems in Sections 3, 4, 6, 7, 8, respectively. Our lower bounds on the average sensitivity of algorithms for the global minimum cut problem and the maximum matching problem can also be found in Sections 4, 8, respectively. We formally define our notion of average sensitivity for linear program solvers and state our main result concerning a stable-on-average algorithm for
solving linear programs in Section 5. We defer the description of our linear program solver and its analysis to Appendix C. We show a linear lower bound for the 2-coloring problem in Section 9. We discuss general properties of average sensitivity in Section 10.

2 Preliminaries

For a positive integer \( n \), let \([n] = \{1, 2, \ldots, n\} \). Let \( G = (V,E) \) be a graph. For a positive integer \( k \leq |E| \), we use the notation \( \binom{E}{k} \) to denote the set of all subsets of \( E \) of cardinality \( k \). For an edge \( e \in E \), we denote by \( G - e \) the graph obtained by removing \( e \) from \( G \). Similarly, for an edge set \( F \subseteq E \), we denote by \( G - F \) the graph obtained by removing every edge in \( F \) from \( G \). For an edge set \( F \subseteq E \), let \( V(F) \) denote the set of vertices incident to an edge in \( F \). For a vertex set \( S \), let \( G[S] \) denote the subgraph of \( G \) induced by \( S \). We often use the symbols \( n \), \( m \), \( \Delta \) to denote the number of vertices, the number of edges, and the maximum degree of a vertex, respectively, in the input graph. We use \( \text{OPT}(G) \) to denote the optimal value of a graph \( G \) in the graph problem we are concerned with. We simply write \( \text{OPT} \) when \( G \) is clear from the context. We denote by \( \mathcal{G} \) the (infinite) set consisting of all graphs. We denote by \( \mathbb{R}_+ \) the set of non-negative real numbers. For a set \( S \subseteq V \), let \( \chi_S \in \mathbb{R}^V \) denote the characteristic vector of \( S \). For vectors \( x, y \in \mathbb{R}^n \), we use \( \langle x, y \rangle \) to denote the inner product of \( x \) and \( y \).

2.1 Exponential Mechanism

The exponential mechanism [24] is an algorithm that, given a vector \( x \in \mathbb{R}^n \) and a real number \( \eta > 0 \), returns an index \( i \in [n] \) with probability proportional to \( e^{-\eta x(i)} \). Just as the exponential mechanism is useful to design differentially private algorithms, it is also useful to design stable-on-average algorithms. Lemma 2.1 formalizes this statement.

Lemma 2.1. Let \( \eta > 0 \) and let \( A \) be the algorithm that, given a vector \( x \in \mathbb{R}^n \), applies the exponential mechanism to \( x \) and \( \eta \). Then for any \( t > 0 \), we have

\[
\operatorname{Pr}_{i \sim A(x)} \left[ x(i) \geq \text{OPT} + \frac{\log n}{\eta} + \frac{t}{\eta} \right] \leq e^{-t},
\]

where \( \text{OPT} = \min_{i \in [n]} x(i) \). Moreover, for all \( x' \in \mathbb{R}^n \), we have

\[
d_{\text{TV}}(A(x), A(x')) = O \left( \eta \cdot \| x - x' \|_1 \right).
\]

The proof of Lemma 2.1 is deferred to Appendix A. By setting \( \eta = \log n / \epsilon \) and replacing \( t \) with \( t \log n \), we get the following:

Lemma 2.2. Let \( \epsilon > 0 \). There exists an algorithm \( A_\epsilon \) such that, given a vector \( x \in \mathbb{R}^n \) outputs \( i \in [n] \) such that

\[
\operatorname{Pr}_{i \sim A_\epsilon(x)} \left[ x(i) \geq \text{OPT} + \epsilon(1 + t) \right] \leq n^{-t},
\]

for any \( t > 0 \), where \( \text{OPT} = \min_{i \in [n]} x(i) \). Moreover, for all \( x' \in \mathbb{R}^n \), we have

\[
d_{\text{TV}}(A_\epsilon(x), A_\epsilon(x')) = O \left( \| x - x' \|_1 \cdot \frac{\log n}{\epsilon} \right).
\]
3 Warm Up: Minimum Spanning Forest

To get intuition about average sensitivity of algorithms, we start with the minimum spanning forest problem. In this problem, we are given a weighted graph \( G = (V, E, w) \), where \( w : E \rightarrow \mathbb{R} \) is a weight function on edges, and we want to find a forest of the minimum total weight including all the vertices.

Recall that Kruskal’s algorithm \([\text{20}]\) works as follows: Iterate over edges in the order of increasing weights, where we break ties arbitrarily. At each iteration, add the current edge to the solution if it does not form a cycle with the edges already added. The following theorem states that this simple and deterministic algorithm is stable-on-average.

**Theorem 3.1.** The average sensitivity of Kruskal’s algorithm is \( O(n/m) \).

**Proof.** Let \( G = (V, E) \) be the input graph and \( T \) be the spanning forest obtained by running Kruskal’s algorithm on \( G \). We consider how the output changes when we remove an edge \( e \in E \) from \( G \).

If the edge \( e \) does not belong to \( T \), clearly the output of Kruskal’s algorithm on \( G - e \) is also \( T \).

Suppose that the edge \( e \) belongs to \( T \). Let \( T_1 \) and \( T_2 \) be the two trees rooted at the endpoints of \( e \) obtained by removing \( e \) from \( T \). If \( G - e \) is not connected, that is, \( e \) is a bridge in \( G \), then Kruskal’s algorithm outputs \( T_1 \cup T_2 \) on \( G - e \). If \( G - e \) is connected, then let \( e' \) be the first edge considered by Kruskal’s algorithm among all the edges connecting \( G[V(T_1)] \) and \( G[V(T_2)] \), where \( V(T_i) \) is the vertex set of \( T_i \) for \( i \in [2] \). Then, Kruskal’s algorithm outputs \( T_1 \cup T_2 \cup \{e'\} \) on \( G - e \).

It follows that the Hamming distance between \( T \) and the output of the algorithm on \( G - e \) is at most 2.

Therefore, the average sensitivity of Kruskal’s algorithm is at most

\[
\frac{m - |T|}{m} \cdot 0 + \frac{|T|}{m} \cdot 2 = O\left(\frac{n}{m}\right).
\]

□

In Appendix \([\text{B}]\) we show that Prim’s algorithm, another classical algorithm for the minimum spanning forest problem, has average sensitivity \( \Omega(m) \) for a certain natural tie breaking rule.

4 Global Minimum Cut

For a graph \( G = (V, E) \) and a vertex set \( S \subseteq V \), we define \( \text{cost}(G, S) \) to be the number of edges in \( E \) that cross the cut \((S, V \setminus S)\). Then in the global minimum cut problem, given a graph \( G = (V, E) \), we want to compute a vertex set \( \emptyset \subset S \subset V \) that minimizes \( \text{cost}(G, S) \). In this section, we discuss upper and lower bounds on the average sensitivity for the global minimum cut problem.

4.1 Upper bound

In this section, we show the following.

**Theorem 4.1.** For \( \varepsilon > 0 \), there exists a polynomial time algorithm for the global minimum cut problem with approximation ratio \( 2 + \varepsilon \) and average sensitivity \( n^{O(1/\varepsilon \text{OPT})} \).

Let \( \text{OPT} \) be the minimum size of a cut in \( G \). Our algorithm enumerates cuts of small size and then output a vertex set \( S \) with probability \( \exp(-\alpha \cdot \text{cost}(G, S)) \) for a suitable \( \alpha \). See Algorithm \([\text{1}]\) for details.
Algorithm 1: Stable Algorithm for Global Minimum Cut

Input: undirected graph $G = (V, E)$, $\varepsilon > 0$

1. Compute the value $\text{OPT}$;
2. Let $\alpha \leftarrow \frac{(2+1/\varepsilon) \log n}{\text{OPT}}$ denote a parameter;
3. Enumerate all cuts of size at most $(2 + 7\varepsilon) \text{OPT} + 2\varepsilon$;
4. Sample a vertex set $S$ (from among the cuts enumerated) with probability proportional to $\exp(-\alpha \cdot \text{cost}(G, S))$;
5. return $S$.

The approximation ratio of the Algorithm 1 is $2 + 9\varepsilon$: It clearly holds when $\text{OPT} \geq 1$, and it also holds when $\text{OPT} = 0$ because we only output a cut of size zero (for $\varepsilon < 1/2$). The following theorem due to Karger [15] directly implies that it runs in time polynomial in the input size for any constant $\varepsilon > 0$.

**Theorem 4.2 ([15]).** Given a graph $G$ on $n$ vertices with the minimum cut size $c$ and a parameter $\alpha \geq 1$, the number of cuts of size at most $\alpha \cdot c$ is at most $n^{2\alpha}$ and can be enumerated in time polynomial (in $n$) per cut.

We now show that Algorithm 1 is stable-on-average.

**Lemma 4.3.** The average sensitivity of Algorithm 1 is at most

$$\beta(G) = \frac{n}{m} \cdot n^{(2+1/\varepsilon)/\text{OPT}} \cdot ((2 + 7\varepsilon) \text{OPT} + 2\varepsilon) + o(1).$$

As we have $\text{OPT} \leq 2m/n$, the average sensitivity can be bounded by $n^{O(1/\varepsilon \text{OPT})}$, and Theorem 4.1 follows by replacing $\varepsilon$ with $\varepsilon/9$.

**Proof.** If $\text{OPT} = 0$, then the claim trivially holds because the right hand size is infinity. Hence in what follows, we assume $\text{OPT} \geq 1$.

Let $\mathcal{A}$ denote Algorithm 1. Consider an (inefficient) algorithm $\mathcal{A}'$ that on input $G$, outputs a cut $S \subseteq V$ (from among all the cuts in $G$) with probability proportional to $\exp(-\alpha \cdot \text{cost}(G, S))$.

For a graph $G = (V, E)$, let $\mathcal{A}(G)$ and $\mathcal{A}'(G)$ denote the output distribution of algorithms $\mathcal{A}$ and $\mathcal{A}'$ on input $G$, respectively. For $G = (V, E)$ and $S \subseteq V$, let $p_G(S)$ and $p'_G(S)$ be shorthands for the probabilities that $S$ is output on input $G$ by algorithms $\mathcal{A}$ and $\mathcal{A}'$, respectively.

We first bound the earth mover’s distance between $\mathcal{A}(G)$ and $\mathcal{A}'(G)$ for a graph $G = (V, E)$. To this end, we define

$$Z = \sum_{S \subseteq V : \text{cost}(G, S) \leq \text{OPT} + b} \exp(-\alpha \cdot \text{cost}(G, S)), \quad \text{and } Z' = \sum_{S \subseteq V} \exp(-\alpha \cdot \text{cost}(G, S))$$

where $b = (1 + 7\varepsilon) \text{OPT} + 2\varepsilon$. Note that $Z \leq Z'$ and the quantity $\frac{Z'}{Z}$ is the total probability mass assigned by algorithm $\mathcal{A}'$ to cuts $S \subseteq V$ such that $\text{cost}(G, S) > \text{OPT} + b$.

Now, we start with $\mathcal{A}'(G)$. For each $S \subseteq V$ such that $\text{cost}(G, S) \leq \text{OPT} + b$, keep at least $\frac{Z'}{Z} \cdot p'_G(S)$ mass with a cost of 0 and move a mass of at most $p'_G(S) - \frac{Z'}{Z} \cdot p'_G(S)$ at a cost of
\[ d_{\text{EM}}(\mathcal{A}(G), \mathcal{A}'(G)) \leq n \cdot \sum_{S \subseteq V : \text{cost}(G, S) \leq \text{OPT} + b} p'_G(S) \left(1 - \frac{Z}{Z'}\right) + n \cdot \sum_{S \subseteq V : \text{cost}(G, S) > \text{OPT} + b} p'_G(S) \]

\[ = \frac{n(Z' - Z)}{Z'} \left(\sum_{S \subseteq V : \text{cost}(G, S) \leq \text{OPT} + b} p'_G(S) + 1\right) = \frac{2n(Z' - Z)}{Z'} \]

Let \( n_t \) stand for the number of cuts of cost at most \( \text{OPT} + t \) in \( G \). By Karger’s theorem (Theorem 4.2), we have that \( n_t \leq n^{2+2t/\text{OPT}} \). Then, we have

\[ \frac{Z' - Z}{Z'} \leq \sum_{t > b} \exp(-\alpha t) \cdot (n_t - n_{t-1}) \leq (\exp(\alpha) - 1) \cdot \sum_{t > b} \exp(-\alpha t)n_t \]

\[ \leq (\exp(\alpha) - 1)n^2 \cdot \sum_{t > b} n^{2t/\text{OPT}} \cdot \exp(-\alpha t) \]

\[ \leq (\exp(\alpha) - 1)n^2 \cdot \sum_{t > b} n^{-t/\text{OPT}} \leq (\exp(\alpha) - 1)n^2 \cdot \frac{n^{-(b+1)/\varepsilon \text{OPT}}}{1 - n^{-1/\varepsilon \text{OPT}}} \]

\[ = \left(n^{(2+1/\varepsilon)/\text{OPT} - 1}\right) \cdot \left(1 + \frac{1}{n^{1/\varepsilon \text{OPT}} - 1}\right) \cdot \frac{n^2}{n^{(b+1)/\varepsilon \text{OPT}}} \]

\[ \leq n^{(2+1/\varepsilon)/\text{OPT}} \cdot \left(1 + \frac{\varepsilon n}{\log n}\right) \cdot \frac{n^2}{n^{(b+1)/\varepsilon \text{OPT}}} \]

\[ = O\left(\frac{\varepsilon n^{3+(2+1/\varepsilon)/\text{OPT}}}{n^{(b+1)/\varepsilon \text{OPT}}}\right) = O\left(\frac{\varepsilon}{n^{1+1/\varepsilon}}\right). \]

The last inequality above follows from our choice of \( b \). Therefore, the earth mover’s distance between \( \mathcal{A}(G) \) and \( \mathcal{A}'(G) \) is 

\[ d_{\text{EM}}(\mathcal{A}(G), \mathcal{A}'(G)) \leq O(\frac{\varepsilon}{n^{1+1/\varepsilon}}). \]

In addition, we can bound the expected size of the cut output by \( \mathcal{A}' \) on \( G \) as follows. The total probability mass assigned by \( \text{Algorithm } A' \) to cuts of size larger than \( \text{OPT} + b \) is equal to 

\[ \frac{Z' - Z}{Z'} = O\left(\frac{\varepsilon}{n^{1+1/\varepsilon}}\right). \]

Hence, the expected size of the cut output by \( \mathcal{A}' \) on \( G \) is at most \( \text{OPT} + b + m \cdot O(\frac{\varepsilon}{n^{1+1/\varepsilon}}) = (2 + 7\varepsilon)\text{OPT} + 2\varepsilon + O(\frac{\varepsilon m}{n^{1+1/\varepsilon}}) \).

We now bound the earth mover’s distance between \( \mathcal{A}'(G) \) and \( \mathcal{A}'(G - e) \) for an arbitrary edge \( e \in E \). Let \( Z'_e \) denote the quantity \( \sum_{S \subseteq V} \exp(-\alpha \cdot \text{cost}(G - e, S)) \). Since the cost of every cut in \( G - e \) is at most the cost of the same cut in \( G \), we have that \( Z' \leq Z'_e \) and therefore,

\[ p'_G(S) = \frac{\exp(-\alpha \cdot \text{cost}(G, S))}{Z'} \leq \frac{\exp(\alpha \cdot \text{cost}(G - e, S))}{Z'_e} \cdot \frac{Z'_e}{Z'} = p'_{G-e}(S) \cdot \frac{Z'_e}{Z'}. \]

We transform \( \mathcal{A}'(G) \) into \( \mathcal{A}'(G - e) \) as follows. For each \( S \subseteq V \), we leave a probability mass of at most \( p'_{G-e}(S) \) at \( S \) with zero cost and move a mass of \( \max\{0, p'_G(S) - p'_{G-e}(S)\} \) to any other point at a cost of at most \( n \cdot \max\{0, p'_G(S) - p'_{G-e}(S)\} \) \( \leq n \cdot \left(\frac{Z'_e}{Z'} - 1\right) \cdot p'_G(S) \). Hence,

\[ d_{\text{EM}}(\mathcal{A}'(G), \mathcal{A}'(G - e)) \leq n \cdot \left(\frac{Z'_e}{Z'} - 1\right) \cdot \sum_{S \subseteq V} p'_G(S) = n \cdot \left(\frac{Z'_e}{Z'} - 1\right). \]
By the triangle inequality, the earth mover’s distance between $A(G)$ and $A(G - e)$ can be bounded as
\[
d_{EM}(A(G), A(G - e)) \leq d_{EM}(A(G), A'(G)) + d_{EM}(A'(G), A'(G - e)) + d_{EM}(A'(G - e), A(G - e))
\]
\[
\leq n \cdot \left( \frac{Z'_G}{Z} - 1 \right) + O\left( \frac{2\varepsilon}{n^{2+1/\varepsilon}} \right).
\]

Hence, the average sensitivity of $A$ is bounded as:
\[
\beta(G) = \mathbb{E}_{e \sim E} d_{EM}(A(G), A(G - e)) \leq O\left( \frac{2\varepsilon}{n^{3+1/\varepsilon}} \right) + n \cdot \mathbb{E}_{e \sim E} \left( \frac{Z'_G}{Z} - 1 \right)
\]
\[
= O\left( \frac{2\varepsilon}{n^{3+1/\varepsilon}} \right) + \frac{n}{mZ'} \sum_{e \in E} (Z'_e - Z')
\]
\[
= O\left( \frac{2\varepsilon}{n^{3+1/\varepsilon}} \right) + \frac{n}{mZ'} \sum_{e \in E} \sum_{V : e \text{ crosses } S} \exp(-\alpha \cdot \text{cost}(G - e, S)) - \exp(-\alpha \cdot \text{cost}(G, S))
\]
\[
= O\left( \frac{2\varepsilon}{n^{3+1/\varepsilon}} \right) + \frac{n(\exp(\alpha) - 1)}{mZ'} \sum_{e \in E} \sum_{V : e \text{ crosses } S} \exp(-\alpha \cdot \text{cost}(G, S))
\]
\[
= O\left( \frac{2\varepsilon}{n^{3+1/\varepsilon}} \right) + \frac{n(\exp(\alpha) - 1)}{m} \sum_{S \subseteq V} \text{cost}(G, S) \cdot \frac{\exp(-\alpha \cdot \text{cost}(G, S))}{Z'}.
\]

The summation in the second term above is equal to the expected size of the cut output by algorithm $A'$ on input $G$. We argued that it is at most $(2 + 7\varepsilon)OPT + 2\varepsilon + O\left( \frac{em}{n^{2+1/\varepsilon}} \right)$. Hence, the average sensitivity of $A$ is at most
\[
\frac{n}{m} \cdot n(2+1/\varepsilon)/OPT \cdot ((2 + 7\varepsilon)OPT + 2\varepsilon) + O\left( \frac{\varepsilon n(2+1/\varepsilon)/OPT + 2}{n^{3+1/\varepsilon}} \right)
\]
\[
= \frac{n}{m} \cdot n(2+1/\varepsilon)/OPT \cdot ((2 + 7\varepsilon)OPT + 2\varepsilon) + o(1)
\]
as $OPT \geq 1$. \hfill \square

### 4.2 Lower bound

In this section, we show that the average sensitivity of the algorithm given in Section 4.1 is almost tight. Specifically, we show the following.

**Theorem 4.4.** Any algorithm for the global minimum cut problem with no additive error (and possibly an arbitrary large multiplicative error) has average sensitivity $\Omega(n^{1/\text{OPT}^2})$ if $\text{OPT} = o(\sqrt{n})$.

**Proof.** We first show a lower bound for the case $\text{OPT} = 1$. Let $A$ be an arbitrary algorithm for the global minimum cut problem with no additive error and let $G = ([n + 1], E)$ be a path on $n + 1$ vertices, where $E = \{(i, i + 1) : i \in [n]\}$. Note that for any $i \in [n]$, the graph $G - (i, i + 1)$ is disconnected and $A$ must output a vertex set $[i]$ or $[n + 1] \setminus [i]$. For a vertex set $S \subseteq [n + 1]$, let $p_S$ be the probability that $A$ on $G$ outputs $S$. Then, the average sensitivity of $A$ on $G$ is
\[
\mathbb{E}_{e \sim E} [d_{EM}(A(G), A(G - e))] = \frac{1}{n} \sum_{i \in [n]} \sum_{S \subseteq [n + 1]} p_S \cdot \min\{d_{\text{Ham}}(S, [i]), d_{\text{Ham}}(S, [n + 1] \setminus [i])\}. \tag{5}
\]
Note that if two sets $S, T \subseteq [n+1]$ satisfy $|S| \leq |T| - n/10$ or $|S| \geq |T| + n/10$, then $d_{\text{Ham}}(S, T) \geq n/10$ holds. Hence, we have $d_{\text{Ham}}(S, [i]) \geq n/10$ for at least a $4/5$-fraction of $i \in [n]$. Similarly, we have $d_{\text{Ham}}(S, [n+1] \setminus [i]) \geq n/10$ for at least a $4/5$-fraction of $i \in [n]$. It follows that we have $\min\{d_{\text{Ham}}(S, [i]), d_{\text{Ham}}(S, [n+1] \setminus [i])\} \geq n/10$ for at least a $3/5$-fraction of $i \in [n]$. Then, we have

$$[5] \geq \frac{3}{5} \cdot \frac{n}{10} \cdot \sum_{S \subseteq [n]} p_S = \frac{3n}{50} = \Omega(n).$$

We now consider the case $t := \text{OPT} \geq 2$. Consider a multigraph $G_t = ([n+1], E_t)$, where $E_t$ contains $t$ copies of the edge $(i, i+1)$ for every $i \in [n]$. For $k = (tn)^{1-1/t}$, the $k$-average sensitivity of $\mathcal{A}$ on $G$ without replacement is

$$\mathbb{E}_{\{e_1, \ldots, e_k\} \sim (E_t^k)} [d_{\text{EM}}(\mathcal{A}(G), \mathcal{A}(G - \{e_1, \ldots, e_k\}))] \geq \mathbb{E}_{\{e_1, \ldots, e_k\} \sim (E_t^k)} [d_{\text{EM}}(\mathcal{A}(G), \mathcal{A}(G - \{e_1, \ldots, e_k\}))] \mid \mathcal{A}(G - \{e_1, \ldots, e_k\}) \text{ has two components}] \times \Pr_{\{e_1, \ldots, e_k\} \sim (E_t^k)} [\mathcal{A}(G - \{e_1, \ldots, e_k\}) \text{ has two components}].$$

The first factor of (7) is exactly equal to (5), which is $\Omega(n)$ by (6). Now we bound the second factor. For every $i \in [n]$, the probability that we cut all the edges between $i$-th and $(i+1)$-th vertices is $\binom{k}{2} / \binom{n}{2}$ from the property of the hypergeometric distribution. For every distinct $i, j \in [n]$, the probability that we cut all the edges between $i$-th and $(i+1)$-th vertices and all the edges between $j$-th and $(j+1)$-th vertices is $\binom{k}{2} / \binom{n}{2}$. By the inclusion-exclusion principle, the probability that $G - \{e_1, \ldots, e_k\}$ has exactly two components is at least

$$n \left( 1 - \frac{t^2}{k} \right) \left( \frac{k}{ln} \right)^t - \left( n \left( \frac{2}{l} \right) \frac{n}{n - 4l} \left( \frac{k}{ln} \right)^{2t} \right) \geq n \left( 1 - \frac{t^2}{k} \right) \left( \frac{k}{ln} \right)^t - \left( n \left( \frac{2}{l} \right) \frac{n}{n - 4l} \frac{1}{2t} \right)^{2t} \geq \frac{3}{4t} - \frac{1}{t^2} \geq \Omega\left( \frac{1}{t} \right),$$

where we used the fact that $2 \leq t = o(\sqrt{n})$. Hence, we have (7) = $\Omega(n/t)$. By Theorem 1.5, the average sensitivity $\beta$ of $\mathcal{A}$ on $G$ must satisfy

$$\beta \cdot (tn)^{1-1/t} \geq \Omega\left( \frac{n}{t} \right),$$

which implies $\beta \geq \Omega(n^{1/t}/t^2)$.

5 Linear Programming

In linear programming (LP), given a cost vector $c \in \mathbb{R}^n_+$ and a polytope $K \subseteq \mathbb{R}^n$, we want to compute a vector $x \in K$ that minimizes $\langle c, x \rangle$. In this section, we show an LP solver with small average sensitivity with respect to the change in $c$, which will be useful to design stable-on-average algorithms for the minimum $s$-$t$ cut and the balanced cut problems.

First we formally define the average sensitivity of an LP solver. We fix a polytope $K \subseteq [0,1]^n$ and let $\mathcal{A}$ be an LP solver over $K$. Given a cost vector $c \in [0,1]^n$, the goal of $\mathcal{A}$ is to find a vector
In this section, we design a stable-on-average algorithm for the cost vector
\(c \in [0,1]^n\). For \(i \in [n]\), let \(c^{-i} \in [0,1]^n\) denote the vector obtained from \(c\) by setting \(c(i) = 0\). The average sensitivity of \(A\) is defined as
\[
\mathbb{E}_{\eta \sim [n]} d_{EM}(A(c), A(c^{-i}))
\]
where \(d_{EM}(A(c), A(c^{-i}))\) is the earth mover’s distance between \(A(c)\) and \(A(c^{-i})\) with the underlying distance function being the \(\ell_1\) distance.

Let \(OPT(c)\) be the optimal value of the linear program \(\min_{x \in K} \langle \eta, x \rangle\), where the polytope
\(K \subseteq [0,1]^n\) should be always clear from the context. Note that \(OPT(c) \geq OPT(c^{-i})\) for every \(i \in [n]\). We simply write \(OPT\) instead of \(OPT(c)\) when \(c\) is clear from the context.

Let \(0 < \eta \leq 1\) be a parameter. We define a Gibbs distribution \(D_{\eta,e,K}\) over the polytope
\(K \subseteq [0,1]^n\) so that the probability density function of \(D_{\eta,e,K}\) is
\(f_{\eta,e,K}(x) := \exp(-\eta \langle c, x \rangle) / Z_{\eta,e,K}\), where
\(Z_{\eta,e,K} = \int_K \exp(-\eta \langle c, x \rangle) dx\). We provide an LP solver with a logarithmic average sensitivity, assuming that we can efficiently and approximately draw a sample from the Gibbs distribution \(D_{\eta,e,K}\).

**Theorem 5.1.** Let \(\epsilon > 0\) and \(K \subseteq [0,1]^n\) be a polytope. Suppose that, for any cost vector \(c \in [0,1]^n\), we can draw a sample from a distribution \(D\) with
\(d_{TV}(D, D_{\eta,e,K}) \leq \epsilon\) in time polynomial in \(n, e^{\eta}\), and \(\log(1/\epsilon)\). Then, there exists a polynomial time algorithm \(A\) that, given a cost vector \(c \in [0,1]^n\), outputs \(x \in K\) with
\[
\mathbb{E}_{x \sim A(c)} \langle c, x \rangle \leq (1 + \epsilon)OPT + O(\log n).
\]
Moreover, the average sensitivity of \(A\) is \(O(\log n/\epsilon)\).

We can use, e.g., the algorithm given in [22] to draw a sample from the distribution \(D\) in Theorem 5.1. We explain our algorithm and its average sensitivity analysis in Section C.

### 6 s-t Minimum Cut

In this section, we design a stable-on-average algorithm for the \(s-t\) minimum cut problem. We say that a pair \(\{u,v\} \in \binom{V}{2}\) is cut by \(S \subseteq V\) if \(u \in S\) and \(v \in V \setminus S\) or vice versa. The cut size of a vertex set \(S \subseteq V\) in a graph \(G = (V,E)\), denoted by \(e_G(S)\), is the number of edges \(e \in E\) cut by \(S\). In the \(s-t\) minimum cut problem, given a graph \(G = (V,E)\) and two vertices \(s,t \in V\), we want to find a minimum \(s-t\) cut, that is, a vertex set \(S\) with \(s \in S\) and \(t \notin S\) that has the minimum cut size. Our goal is to show the following.

**Theorem 6.1.** For each \(\epsilon > 0\), there exists a polynomial time \((1 + \epsilon, O(\log n))\)-approximation algorithm with average sensitivity \(O(\log n/\epsilon)\) for the minimum \(s-t\) cut problem.

We describe our algorithm in Section 6.1 and analyze it in Section 6.2.

#### 6.1 Algorithm

Our algorithm is based on an LP relaxation for the minimum \(s-t\) cut problem. For each pair of vertices \(\{u,v\} \in \binom{V}{2}\), we introduce a variable \(d(\{u,v\})\), which we regard as a distance between \(u\) and \(v\). Roughly speaking, \(d(\{u,v\}) = 1\) if \(u\) and \(v\) are on different sides of an \(s-t\) cut. \(d(\{u,v\})\) is
Algorithm 2: Algorithm for the minimum $s$-$t$ cut problem

1 Procedure MinCut($G, s, t, \epsilon$)
   
   **Input:** A graph $G = (V, E)$, two vertices $s, t \in V$, and $\epsilon > 0$.

2 Solve LP \eqref{lp} using the algorithm given by Theorem 5.1 with parameter $\epsilon > 0$ and let $d \in [0, 1]^{\binom{V}{2}}$ be the solution obtained;

3 $z \leftarrow (d(s, v))_{v \in V}$;

4 return Thresh($z$).

5 Procedure Thresh($x$)

   **Input:** $x \in [0, 1]^V$

6 Sample $\tau$ from $[0, 1]$ uniformly at random;

7 return the set $\{v \in V : x(v) \geq \tau\}$

0 otherwise. For notational simplicity, we often write $d(u, v)$ to denote $d(\{u, v\})$. Intuitively, the distance between $s$ and $t$ should be at least one, and the distance $d(\cdot, \cdot)$ should satisfy the triangle inequality. Our LP relaxation is the following.

\[
\begin{align*}
\text{minimize} & \quad \sum_{\{u,v\} \in E} d(u, v) \\
\text{subject to} & \quad d(s, t) = 1 \\
& \quad d(u, v) + d(v, w) \geq d(u, w) \quad \forall \{u, v, w\} \in \binom{V}{3} \\
& \quad 0 \leq d(u, v) \leq 1 \quad \forall \{u, v\} \in \binom{V}{2}
\end{align*}
\] \(\text{(8)}\)

It is easy to check that LP \eqref{lp} is indeed a relaxation for the minimum $s$-$t$ cut problem. Let $S \subseteq V$ be a minimum $s$-$t$ cut. Then for each $\{u, v\} \in E$, we set $d(u, v) = 1$ if $\{u, v\}$ is cut by $S$, and set $d(u, v) = 0$ otherwise. It is clear that $\sum_{\{u,v\} \in E} d(u, v) = |\{\{u, v\} \in E : \{u, v\} \text{ is cut by } S\}|$ is the cut size of $S$ and that $d$ satisfies all the constraints.

Our algorithm is given in Algorithm 2. It simply computes a solution $d \in [0, 1]^{\binom{V}{2}}$ to LP \eqref{lp} using the algorithm given in Theorem 5.1. We will argue that the polytope given by the constraints in LP \eqref{lp} satisfies the assumption of Theorem 5.1 in Appendix D. Then we round the vector $z := (d(s, v))_{v \in V} \in [0, 1]^V$ using the procedure Thresh, which is based on randomized thresholding.

For $x \in [0, 1]^V$, let $v_1, \ldots, v_n$ be an ordering of $V$ such that $x(v_i) \geq x(v_{i+1})$ for every $i \in [n - 1]$. Note that Thresh($x$) outputs the set $\{v_1, \ldots, v_i\}$ with probability $x(v_i) - x(v_{i+1})$ for $i \in [n + 1]$, where we define $x(v_{n+1}) = 0$ for a dummy vertex $v_{n+1}$.

6.2 Analysis

In this section, we analyze Algorithm 2. First, we analyze the solution quality of Thresh.

**Lemma 6.2.** We have

$$
\mathbb{E}[e_G(\text{Thresh}(z))] \leq \sum_{e \in E} d(e),
$$

where $z = (d(s, v))_{v \in V}$.
Proof. For each edge \( e \in E \), the probability that it is cut by \( S \) is
\[
|z(u) - z(v)| = |d(s, u) - d(s, v)| \leq d(u, v),
\]
where \( u, v \in V \) are the endpoints of \( V \). The claim follows by the linearity of expectations.

Now, we bound the average sensitivity of \( \text{Thresh} \) when only one coordinate differs.

**Lemma 6.3.** Let \( x \in [0, 1]^V \) and \( x' \in [0, 1]^V \) be such that
\[
x'(u) = \begin{cases} 
    x(u) + \Delta & \text{if } u = v, \\
    x(u) & \text{otherwise},
\end{cases}
\]
for some \( v \in V \) and \( 0 \leq x(v) \leq 1 - \Delta \). Then, \( d_{EM}(\text{Thresh}(x), \text{Thresh}(x')) \leq 2\Delta \).

Proof. We can assume \( \Delta \geq 0 \) as otherwise we can switch the roles of \( x \) and \( x' \). Starting with the vector \( x_0 = x \), we iteratively construct \( x_k \in [0, 1]^V \) from \( x_{k-1} \) as
\[
x_k(u) = \begin{cases} 
    \min \{ x'(v), \min_{u \in V: x_{k-1}(u) > x_{k-1}(v)} \{ x_{k-1}(u) \} \} & \text{if } u = v, \\
    x_{k-1}(u) & \text{otherwise},
\end{cases}
\]
Let \( \ell \) be the smallest integer such that \( x_{\ell}(v) = x'(v) \). Note that for every \( k \in [\ell] \), there is an ordering \( v_1, \ldots, v_n \) of \( V \) such that both \( x_k(v_i) \geq x_k(v_{i+1}) \) hold for every \( i \in [n-1] \). Note that \( \sum_{k \in [\ell]} \|x_k - x_{k-1}\|_1 = \Delta \).

Now we show that for each \( k \in [\ell] \) we have \( d_{EM}(\text{Thresh}(x_k), \text{Thresh}(x_{k-1})) \leq 2\|x_k - x_{k-1}\|_1 \).

Let \( v_1, \ldots, v_n \) be an ordering of \( V \) with the property mentioned above, and let \( S = \{v_1, \ldots, v_{i-1}\} \) and \( S' = \{v_1, \ldots, v_i\} \), where \( i \in [n] \) is such that \( v_i = v \). Then, the only difference in the output distributions of \( \text{Thresh}(x_k) \) and \( \text{Thresh}(x_{k-1}) \) is that the former outputs \( S \) with probability \( x_k(v_{i-1}) - x_k(v_i) \) and \( S' \) with probability \( x_k(v_i) - x_k(v_{i+1}) \) whereas the latter outputs \( S \) with probability \( x_{k-1}(v_{i-1}) - x_{k-1}(v_i) \) and \( S' \) with probability \( x_{k-1}(v_i) - x_{k-1}(v_{i+1}) \). It follows that
\[
d_{EM}(\text{Thresh}(x_k), \text{Thresh}(x_{k-1})) \leq 2(x_k(v_i) - x_{k-1}(v_i)) \cdot d_{\text{Ham}}(S, S') = 2\|x_k - x_{k-1}\|_1.
\]

Then, we have
\[
d_{EM}(\text{Thresh}(x), \text{Thresh}(x')) = d_{EM}(\text{Thresh}(x_0), \text{Thresh}(x_{\ell}))
\leq \sum_{k \in [\ell]} d_{EM}(\text{Thresh}(x_k), \text{Thresh}(x_{k-1}))
\leq \sum_{k \in [\ell]} 2\|x_k - x_{k-1}\|_1 = 2\Delta,
\]
as desired.

**Corollary 6.4.** \( d_{EM}(\text{Thresh}(x), \text{Thresh}(x')) \leq 2\|x - x'\|_1 \) for \( x, x' \in [0, 1]^V \).
Proof. The inequality can be obtained by iteratively applying Lemma 6.3 to each coordinate of the vectors.

Proof of Theorem 6.1. Let $A$ be Algorithm 2. First, we consider the approximation guarantee of $A$. Let $d^* \in \mathbb{R}^{(V)}_{+}$ be an optimal solution to LP (8). By Theorem 5.1, we get $d \in [0,1]^{(V)}_{+}$ satisfying the constraints in LP (8) such that

$$
\mathbb{E}_{d \sim A} \left[ \sum_{\{u,v\} \in E} d(u,v) \right] \leq (1 + \epsilon) \sum_{\{u,v\} \in E} d^*(u,v) + O(\log n).
$$

By Lemma 6.2, the expected cut size of the output set is the same.

Now, we consider the average sensitivity of $A$. Let $G = (V,E)$ be a graph, and let $d \in [0,1]^{(V)}_{+}$ be the solution to LP (8) computed by $A$ on $G$. For each edge $e \in E$, let $d_e \in [0,1]^{(V)}_{+}$ be the solution to LP (8) computed by $A$ on $G - e$. Note that $d$ and $d_e$ ($e \in E$) are random variables. For $e \in E$, let $\mu_e : [0,1]^{(V)}_{+} \times [0,1]^{(V)}_{+} \rightarrow \mathbb{R}_+$ be a joint distribution that attains

$$
d_{e1}^E(d,d_e) = \int \|d - d_e\| \mu_e(d,d_e) d(d,d_e).
$$

Then, we have

$$
\mathbb{E}_{e \sim E} d_{e1}^E(A(G),A(G - e)) \\
\leq \mathbb{E}_{e \sim E} \int d_{e1}^E \left( \text{THRESH}((d(s,v))_{v \in V}), \text{THRESH}((d_e(s,v))_{v \in V}) \right) \mu_e(d,d_e) d(d,d_e) \\
\leq 2 \mathbb{E}_{e \sim E} \int \|d - d_e\| \mu_e(d,d_e) d(d,d_e) \quad \text{(by Corollary 6.4)} \\
= 2 \mathbb{E}_{e \sim E} d_{e1}^E(d,d_e) \\
= O \left( \frac{\log n}{\epsilon} \right), \quad \text{(by Theorem 5.1)}
$$

as desired.

7 Balanced Cut

In the balanced cut problem, given a graph $G = (V,E)$ and $0 < \alpha < 1/2$, we want to compute a vertex set $\emptyset \subsetneq S \subsetneq V$ that minimizes the number of edges $e(S,V \setminus S)$ between $S$ and $V \setminus S$ subject to $\alpha n \leq |S| \leq (1 - \alpha)n$. Leighton and Rao [21], in their seminal work, showed a polynomial time $O(\log n)$-approximation algorithm for a related problem called the sparsest cut problem. In this section, we show a stable-on-average algorithm for the balanced cut problem by modifying the algorithm of Leighton and Rao.

Theorem 7.1. There exists a polynomial time algorithm for the balanced cut problem that, for all $\alpha \in (0,\frac{1}{2})$, outputs a vertex set $S$ with expected cut value

$$
O \left( \frac{\log^4(n) \cdot \log^2(\log n)}{\alpha} \right) \cdot \text{OPT} + O \left( \frac{\log^5(n) \cdot \log^2(\log n)}{\alpha} \right)
$$
Algorithm 3: Bourgain’s embedding

1 Procedure EMBEDDING($d$)

Input: A pseudometric $d \in \mathbb{R}^{V(2)}$ over a set $V$ of cardinality $n$.

2 $U \leftarrow \emptyset$;

3 for $t = 1$ to $\log n$ do

4 for $\Theta(\log n)$ times do

5 $U \leftarrow \emptyset$;

6 Add each $v \in V$ with probability $1/2^t$;

7 $U \leftarrow U \cup \{U\}$.

8 return $f : v \mapsto (f_U(v) : U \in U)$.

and $\rho^4n/16 \leq |S| \leq (1 - \rho^4/16)n$ for $\rho = \Omega\left(\frac{\alpha}{\log^3(n)\log^2(\log n)}\right)$. The average sensitivity of the algorithm is $O\left(\frac{\log^3(n)\log^2(\log n)}{\alpha^2}\right)$.

7.1 LP relaxation

We start with describing the original LP relaxation. First, note that a cut $S \subseteq V$ forms a pseudometric $d_S$ over $V$ defined as

$$d_S(u, v) = \begin{cases} 1 & u \in S \text{ and } v \in V \setminus S, \text{ or } u \in V \setminus S \text{ and } v \in S, \\ 0 & \text{otherwise.} \end{cases}$$

Note that the condition $\alpha n \leq |S| \leq (1 - \alpha)n$ can be rephrased as to $\sum_{\{u, v\} \in V(2)} d_S(u, v) \geq \alpha(1 - \alpha)n^2$. Then, we can relax the condition that pseudometric is given through a cut, and we get the following LP relaxation.

$$\begin{align*}
\text{minimize} & \quad \sum_{e \in E} d(e) \\
\text{subject to} & \quad \sum_{\{u, v\} \in V(2)} d(u, v) \geq \alpha(1 - \alpha)n^2, \\
& \quad d(u, v) + d(v, w) \geq d(u, w) \quad \forall \{u, w\} \in V(3) \\
& \quad 0 \leq d(u, v) \leq 1 \quad \forall \{u, v\} \in V(2) \tag{9}
\end{align*}$$

We denote by LP the optimal value of LP (9). We will solve LP (9) using Theorem 5.1 and we will argue that the polytope given by the constraints in LP (9) satisfies the assumption of Theorem 5.1 in Appendix D.

7.2 Embedding

Following Bourgain’s theorem is a crucial component in the $O(\log n)$-approximation algorithm due to Leighton and Rao [21], which constructs a vector representation of vertices in a graph from a pseudometric over the vertex set.
Theorem 7.2 (Bourgain’s theorem [4]). There is a polynomial time algorithm that, given a pseudometric \( d \in \mathbb{R}^{(V)} \) over a set \( V \) of cardinality \( n \), computes a mapping \( f : V \to \mathbb{R}^k \) with \( k = O(\log^2 n) \) such that, with probability at least \( 1 - 1/n \), for every two elements \( u, v \in V \),

\[
\|f(u) - f(v)\|_1 \leq d(u, v) = O \left( \|f(u) - f(v)\|_1 \cdot \log n \right).
\]

Bourgain’s embedding is very simple. For a pseudometric \( d \in \mathbb{R}^{(V)} \) over a vertex set \( V \) and a nonempty subset \( U \subseteq V \), we define \( f_U : V \to \mathbb{R} \) as

\[
f_U(v) = \min_{r \in U} d(r, v).
\]

(10)

For an integer \( t \), let \( \mathcal{D}_t \) be a distribution over subsets of \( V \). A set \( U \subseteq V \) is sampled from \( \mathcal{D}_t \) by adding each vertex \( v \in V \) to \( U \) with probability \( 1/2^t \) independently of other vertices. To construct Bourgain’s embedding, for each \( t = 1, \ldots, \log n \), we sample \( O(\log n) \) sets \( U \) from \( \mathcal{D}_t \), and then use \( f_U(v) \) as one of the coordinates of the vector representing \( v \). A pseudocode of Bourgain’s embedding is given in Algorithm 3.

We first show that the EMBEDDING procedure is stable-on-average. Given a function \( f : V \to \mathbb{R}^k \), for each \( i \in [k] \), we define \( f_i : V \to \mathbb{R} \) to be the function that maps \( v \in V \) to \( f(v)(i) \). Our goal is to show the following.

Lemma 7.3. Let \( d, d' \in \mathbb{R}^{(V)} \) be pseudometrics over a set \( V \) of cardinality \( n \). We have

\[
d_{EM}^f(\text{EMBEDDING}(d), \text{EMBEDDING}(d')) = O \left( \log^2 n \cdot \|d - d'\|_1 \right).
\]

Proof. Let \( U \subseteq V \) be a nonempty subset of vertices. Let \( f'_U \) be the function (10) defined with \( d' \) and \( U \). We first show that

\[
\sum_{v \in V} |f_U(v) - f'_U(v)| = O \left( \|d - d'\|_1 \right),
\]

For a vertex \( v \in V \), let \( r(v) \) (resp., \( r'(v) \)) denote the point in \( U \) closest to \( v \) with respect to the pseudometric \( d \) (resp., \( d' \)). We break ties arbitrarily. Let \( S = \{v \in V : d(r(v), v) \geq d'(r'(v), v)\} \) and \( T = V \setminus S \). Then, we have

\[
\sum_{v \in V} |f_U(v) - f'_U(v)| = \sum_{v \in V} |d(r(v), v) - d'(r'(v), v)|
\]

\[
= \sum_{v \in S} \left( d(r(v), v) - d'(r'(v), v) \right) + \sum_{v \in T} \left( d'(r'(v), v) - d(r(v), v) \right)
\]

\[
\leq \sum_{v \in S} \left( d(r'(v), v) - d'(r'(v), v) \right) + \sum_{v \in T} \left( d'(r(v), v) - d(r(v), v) \right) \quad (\because r(v) \text{ is the point closest to } v \text{ w.r.t. } d)
\]

\[
\leq 2 \sum_{\{u,v\} \in \binom{V}{2}} |d(u, v) - d'(u, v)| = 2 \|d - d'\|_1.
\]

For a multiset \( \mathcal{U} \) of subsets of \( V \), it follows that

\[
\sum_{U \in \mathcal{U}} \sum_{v \in V} |f_U(v) - f'_U(v)| \leq 2 \cdot |\mathcal{U}| \cdot \|d - d'\|_1.
\]

23
Algorithm 4: Algorithm for the balanced cut problem

1 Procedure ROUNDING($f, \alpha$)
   
   **Input:** A function $f : V \rightarrow \mathbb{R}^k$, and $\alpha > 0$.
2 Let $n \leftarrow |V|$;
3 Define $x \in \mathbb{R}^k$ as $x(i) = \sum_{\{u,v\} \in E} |f_i(u) - f_i(v)|$;
4 Choose $j \in [k]$ using the algorithm given by Lemma 2.2 on $-x$ with the setting
   $\epsilon \leftarrow \frac{\alpha(1-\alpha)n^2}{10 C k \log n}$ for a large constant $C$;
5 $\mu_j \leftarrow \frac{1}{n} \sum_{v \in V} f_j(v)$ and $\rho \leftarrow \frac{\alpha(1-\alpha)}{9 C k \log n \log k}$;
6 Choose $t \in [\mu_j - \rho^2/8, \mu_j + \rho^2/8]$ uniformly at random;
7 $S \leftarrow \{v \in V : f_j(v) \leq t\}$;
8 return $S$.

Note that the probability that the procedure EMBEDDING($\cdot$) samples the family $\mathcal{U}$ does not depend on the pseudometric that is given as its input. Let $p_{\mathcal{U}}$ denote this probability. Then,

$$d_{\text{EM}}(\text{EM}(d), \text{EM}(d')) \leq \sum_{\text{multisets } \mathcal{U}} p_{\mathcal{U}} \cdot 2 \cdot |\mathcal{U}| \cdot \|d - d'\|_1$$

$$= O \left( \log^2 n \cdot \|d - d'\|_1 \right) \sum_{\text{multisets } \mathcal{U}} p_{\mathcal{U}}$$

$$= O \left( \log^2 n \cdot \|d - d'\|_1 \right).$$

The first equality above follows from the fact that the dimension of the output of the EMBEDDING procedure is $O(\log^2 n)$.

\[ \square \]

7.3 Rounding

The goal of this section is to provide a rounding procedure with the following guarantee.

**Lemma 7.4.** For any $0 < \alpha < 1/2$ and $\epsilon > 0$, there exists a polynomial time algorithm $A$ that, given a function $f : V \rightarrow \mathbb{R}^k$ such that $\sum_{\{u,v\} \in E} |f(u) - f(v)|_1 \geq \frac{\alpha(1-\alpha)n^2}{C \log n \log k}$ for some $C > 0$, outputs, with probability at least $1 - 1/n^4$, a set $S \subseteq V$ such that

$$\mathbb{E} [e(S, V \setminus S)] = O \left( \frac{1}{\rho^2} \sum_{\{u,v\} \in E} \|f(u) - f(v)\|_1 \right).$$

and $\rho^4 n/16 \leq |S| \leq (1 - \rho^4/16)n$, where $\rho = \frac{\alpha(1-\alpha)}{9 C k \log k \log n}$. Moreover, for $f' : V \rightarrow \mathbb{R}^k$,

$$d_{\text{EM}}(A(f, \alpha), A(f', \alpha)) = O \left( \frac{1}{\rho^2} \sum_{v \in V} \|f(v) - f'(v)\|_1 \right).$$

We describe our rounding procedure. Let $x \in \mathbb{R}^k$ be a vector defined as $x(i) = \sum_{\{u,v\} \in E} |f_i(u) - f_i(v)|$. From the assumption of Lemma 7.4 there exists $i \in [k]$ such that $x(i) \geq \alpha(1-\alpha)n^2/C k \log k \log n$. 24
Lemma 7.5. We have

\[ \Pr_{j} \left[ \min_{t \in [\mu_{j} - \rho^{2}/8, \mu_{j} + \rho^{2}/8]} h_{j}(t) \leq \frac{\rho^{4}n^{2}}{16} \right] \leq \frac{1}{n^{4}} \]

Proof. Applying Lemma 2.2 with the setting \( t = \frac{4 \log n}{\log k} \), we have

\[ \Pr_{j} \left[ x(j) \leq \frac{\alpha(1 - \alpha)n^{2}}{Ck \log k \log n} - \frac{\alpha(1 - \alpha)n^{2}}{10Ck \log^{2} n} \cdot \left( 1 + \frac{4 \log n}{\log k} \right) \right] \leq \frac{1}{k^{2}} \]

\[ \Leftrightarrow \Pr_{j} \left[ x(j) \leq \frac{\alpha(1 - \alpha)n^{2}}{Ck \log n \log k} - \frac{\alpha(1 - \alpha)n^{2}}{10Ck \log^{2} n \log k} \cdot \frac{5 \log n}{\log k} \right] \leq \frac{1}{n^{4}} \]

\[ \Leftrightarrow \Pr_{j} \left[ \mathbb{E}_{t \sim [0,1]} h_{j}(t) \leq \frac{\alpha(1 - \alpha)n^{2}}{2Ck \log k \log n} \right] \leq \frac{1}{n^{4}}. \quad \text{(by 11)} \]

In the remainder of the proof, we condition on the event that \( j \in [k] \) satisfies \( \mathbb{E}_{t \sim [0,1]} h_{j}(t) \geq \frac{\alpha(1 - \alpha)n^{2}}{2Ck \log k \log n} \). Let \( \gamma_{l}, \gamma_{m}, \gamma_{r} \) be the fractions of vertices \( v \in V \) with \( f_{j}(v) \in [0, \mu_{j} - \rho] \), \( f_{j}(v) \in [\mu_{j} - \rho, \mu_{j} + \rho] \), and \( f_{j}(v) \in [\mu_{j} + \rho, 1] \), respectively.

Claim 7.6. At least one of \( \gamma_{l} \) and \( \gamma_{r} \) is larger than \( \rho/2 \).

Proof. Suppose for the sake of contradiction that \( \gamma_{l} \leq \rho/2 \) and \( \gamma_{r} \leq \rho/2 \). Then, we have

\[ \frac{\alpha(1 - \alpha)n^{2}}{2Ck \log k \log n} \leq \mathbb{E}_{t \sim [0,1]} h_{j}(t) = \sum_{\{u,v\} \in \binom{V}{2}} |f_{j}(u) - f_{j}(v)| \leq \gamma_{l}n^{2} + 2\rho \gamma_{m}n^{2} + \gamma_{r}n^{2} \leq 3\rho n^{2} = \frac{\alpha(1 - \alpha)n^{2}}{3Ck \log k \log n}, \]

which is a contradiction.

Let \( \tilde{\gamma}_{l}, \tilde{\gamma}_{m}, \tilde{\gamma}_{r} \) be the fractions of vertices \( v \in V \) with \( f_{j}(v) \in [0, \mu_{j} - \rho^{2}/8] \), \( f_{j}(v) \in [\mu_{j} - \rho^{2}/8, \mu_{j} + \rho^{2}/8] \), and \( f_{j}(v) \in [\mu_{j} + \rho^{2}/8, 1] \), respectively. Note that \( \tilde{\gamma}_{l} \geq \gamma_{l}, \tilde{\gamma}_{m} \leq \gamma_{m} \) and \( \tilde{\gamma}_{r} \geq \gamma_{r} \).

Claim 7.7. Both \( \tilde{\gamma}_{l} \) and \( \tilde{\gamma}_{r} \) are no less than \( \rho^{2}/4 \).
**Proof.** Suppose for the sake of contradiction that $\tilde{\gamma}_l \leq \rho^2/4$ or $\tilde{\gamma}_r \leq \rho^2/4$. Here, we consider the case $\tilde{\gamma}_l \leq \rho^2/4$. The other case can be handled similarly. By Claim 7.6, at least one of $\gamma_l$ and $\gamma_r$ is larger than $\rho/2$. From our assumption, $\gamma_l \leq \tilde{\gamma}_l \leq \rho^2/4$, and hence we have $\gamma_r \geq \rho/2$.

For the ranges $[0, \mu_j - \rho], (\mu_j, \mu_j - \rho^2/8], (\mu_j - \rho^2/8, \mu_j + \rho^2/8], (\mu_j + \rho^2/8, \mu_j + \rho], (\mu_j + \rho, 1]$, we lower bound the value of $h_j(t)$ by $0, \mu_j - \rho, \mu_j - \rho^2/8, \mu_j + \rho^2/8,$ and $\mu_j + \rho$, respectively. Hence, $\mu_j$ can be lower bounded as:

$$(\tilde{\gamma}_l - \gamma_l) \cdot (\mu_j - \rho) + \tilde{\gamma}_m \cdot (\mu_j - \rho^2/8) + (\tilde{\gamma}_r - \gamma_r) \cdot (\mu_j + \rho^2/8) + \gamma_r \cdot (\mu_j + \rho)$$

$$= (\tilde{\gamma}_l + \tilde{\gamma}_m + \tilde{\gamma}_r) \cdot \mu_j - \gamma_l \cdot \rho - (\mu_j - \tilde{\gamma}_m) \cdot \rho^2/8 - \gamma_r \cdot \mu_j + (\tilde{\gamma}_r - \gamma_r) \cdot \rho^2/8 + \gamma_r \cdot (\mu_j + \rho)$$

$$\geq \mu_j - \tilde{\gamma}_l - \mu_j - \tilde{\gamma}_m \cdot \rho^2/8 + \tilde{\gamma}_r \cdot \rho \cdot \rho^2/8 + \gamma_r \cdot \rho$$

From the above, we obtain

$$-\tilde{\gamma}_l - \frac{\rho^2}{8} \cdot \tilde{\gamma}_m + \frac{\rho^2}{8} (\tilde{\gamma}_r - \gamma_r) + \rho \gamma_r \leq 0,$$

which implies

$$\gamma_m \geq \frac{8}{\rho^2} \left( \rho \gamma_r + \frac{\rho^2}{8} (\tilde{\gamma}_r - \gamma_r) - \tilde{\gamma}_l \right) \geq \frac{8}{\rho^2} \cdot \frac{\rho^2}{4} = 2,$$

which is a contradiction. □

By Claim 7.7, we have

$$\min_{t \in [\mu_j - \rho^2/8, \mu_j + \rho^2/8]} h_j(t) \geq \tilde{\gamma}_l \tilde{\gamma}_r n^2 = \frac{\rho^4 n^2}{16}. \quad \square$$

**Proof of Lemma 7.4.** The expected cut value is

$$\mathbb{E}[e(S, V \setminus S)] = \mathbb{E} \left[ \sum_{j \in [0, \mu_j - \rho^2/8, \mu_j + \rho^2/8]} g_j(t) \right] \leq \mathbb{E} \left[ \frac{4}{\rho^2} \mathbb{E}_{t \sim [0, 1]} g_j(t) \right]$$

$$\leq \mathbb{E} \left[ \frac{4}{\rho^2} \sum_{\{u, v\} \in E} |f_j(u) - f_j(v)| \right] = O \left( \frac{1}{\rho^2} \sum_{\{u, v\} \in E} \|f(u) - f(v)\|_1 \right).$$

Next, we analyze the size of the output set $S$. Note that

$$|S| \cdot (n - |S|) \geq \min_{t \in [\mu_j - \rho^2/8, \mu_j + \rho^2/8]} h_j(t).$$

Hence by Lemma 7.5 with probability at least $1 - 1/n^4$ over the choice of $j$, we have

$$|S| \cdot (n - |S|) \geq \frac{\rho^4 n^2}{16},$$

which implies $\rho^4 n/16 \leq |S| \leq (1 - \rho^4/16)n$.

Let $f': V \to \mathbb{R}^k$ be another embedding. Let $x'$ be the vector constructed by Algorithm 3 from $f'$ and $\alpha$. We now analyze $d_{EM}(A(f, \alpha), A(f', \alpha))$. 

26
To do this, we view $A$ as being composed of two algorithms $A_0$ and $A_1$. The algorithm $A_0$ takes $f : V \rightarrow \mathbb{R}^k, \alpha > 0$ as inputs and outputs $j \in [k]$. The algorithm $A_1$ takes $f, \alpha$, and $A_0(f, \alpha)$ as inputs and outputs the set $S \subseteq V$.

By Lemma 2.2

$$d_{TV}(A_0(x), A_0(x')) = \|x - x'\|_1 \cdot \frac{10Ck \log k \log^2 n}{\alpha(1 - \alpha)n^2}.$$ 

The following claim bounds $\|x - x'\|_1$.

**Claim 7.8.** For $f, f' : V \rightarrow \mathbb{R}^k$ and vectors $x, x' \in \mathbb{R}^k$ constructed by Algorithm 3 from $f$ and $f'$ respectively, we have

$$\|x - x'\|_1 \leq n \cdot \sum_{v \in V} \|f(v) - f'(v)\|_1.$$

**Proof.**

$$\|x - x'\|_1 = \sum_{i \in [k]} |x(i) - x'(i)| = \sum_{i \in [k]} \left| \sum_{(u,v) \in \binom{V}{2}} \left( |f_i(u) - f_i(v)| - |f'_i(u) - f'_i(v)| \right) \right|$$

$$\leq \sum_{i \in [k]} \left| \sum_{(u,v) \in \binom{V}{2}} \left| f_i(u) - f_i(v) - f'_i(u) + f'_i(v) \right| \right| \leq \sum_{i \in [k]} \sum_{(u,v) \in \binom{V}{2}} \|f_i(u) - f'_i(u) + f'_i(v) - f_i(v)\|$$

$$\leq \sum_{i \in [k]} \sum_{(u,v) \in \binom{V}{2}} |f_i(u) - f'_i(u)| + |f'_i(v) - f_i(v)| \leq n \cdot \sum_{v \in V} \|f(v) - f'(v)\|_1. \quad \Box$$

Therefore, we have,

$$d_{TV}(A_0(x), A_0(x')) = O\left( \sum_{v \in V} \|f(v) - f'(v)\|_1 \cdot \frac{k \log k \log^2 n}{n} \right).$$

Next, fix $j \in [k]$. Let $S_{j,t}$ and $S'_{j,t,v}$ denote $A_1(f, \alpha, j)$ and $A_1(f', \alpha, j)$, respectively. We bound $d_{EM}(S_{j,t}, S'_{j,t,v})$. Let $\mu_j$ denote $\frac{1}{n} \sum_{v \in V} f_j(v)$. Note that

$$|\mu_j - \mu'_j| = \left| \frac{1}{n} \sum_{v \in V} f_j(v) - \frac{1}{n} \sum_{v \in V} f'_j(v) \right| \leq \frac{1}{n} \sum_{v \in V} |f_j(v) - f'_j(v)| = O\left( \frac{1}{n} \sum_{v \in V} \|f(v) - f'(v)\|_1 \right).$$

Then,

$$d_{EM}(S_{j,t}, S'_{j,t,v}) \leq 2|\mu_j - \mu'_j| + \frac{E_{t \sim [\rho^2, \rho^2 / 8]} |S_{j,t} \bigtriangleup S'_{j,t}|}{\rho^2} \leq O\left( \sum_{v \in V} \|f(v) - f'(v)\|_1 \right) + \frac{4}{\rho^2} \frac{E_{t \sim [0, 1]} |S_{j,t} \bigtriangleup S'_{j,t}|}{\rho^2} = O\left( \frac{1}{\rho^2} \sum_{v \in V} \|f(v) - f'(v)\|_1 \right).$$

Using our sequential composition theorem, we have:

$$d_{EM}(A(f, \alpha), A(f', \alpha)) \leq n \cdot d_{TV}(A_0(f, \alpha), A_0(f', \alpha)) + \frac{E_{j \in [k]} d_{EM}(A_1(f, \alpha, j), A_0(f', \alpha, j))}{\rho^2} = O\left( \frac{1}{\rho^2} \sum_{v \in V} \|f(v) - f'(v)\|_1 \right). \quad \Box$$

27
Algorithm 5: Algorithm for the balanced cut problem

1 Procedure \textsc{BalancedCut}(G, \alpha)
2 \hspace{1em} \textbf{Input:} A graph $G = (V, E)$ and $\alpha > 0$.
3 \hspace{1em} Solve LP \textsc{[9]} using the algorithm in Theorem \textsc{[5.1]} with $\epsilon = 1$ and let $d \in [0, 1]^{|V|}$ be the obtained solution;
4 \hspace{1em} $f \leftarrow \textsc{Embedding}(d)$;
5 \hspace{1em} $S \leftarrow \textsc{Rounding}(f, \alpha)$;
6 \hspace{1em} return $S$.

7.4 Putting things together

Our algorithm is given in Algorithm 5.

Proof of Theorem 7.1. First, we analyze the approximation guarantee. By Theorem 5.1, we have

$$\sum_{(u,v) \in E} d(u,v) \leq 2\text{LP} + O(\log n).$$

By Theorem 7.2, we obtain $f: V \to \mathbb{R}^k$ with $k = O(\log 2n)$ such that

$$\|f(u) - f(v)\|_1 \leq d(u,v) = O\left(\frac{\log 4n \cdot \log 2 \log n}{\alpha^2} \cdot \text{LP} + \frac{\log 5n \cdot \log 2^2 \log n}{\alpha^2}\right) \quad \text{by Lemma 7.4}$$

with probability at least $1 - 1/n$. In what follows, we condition on the above event having happened.

By Theorem 7.4, we obtain a set $S \subseteq V$ with

$$\mathbb{E}[e(S, V \setminus S)] = O\left(\frac{1}{\rho^2} \sum_{(u,v) \in E} \|f(u) - f(v)\|_1\right)$$

$$= O\left(\frac{\log 4n \cdot \log 2 \log n}{\alpha^2} \sum_{(u,v) \in E} d(u,v)\right) = O\left(\frac{\log 4n \cdot \log 5 \log 2n}{\alpha^2} \cdot \text{LP} + \frac{\log 5n \cdot \log 2^2 \log n}{\alpha^2}\right).$$

Next, we analyze the size of the output set. When (12) holds, we have $\rho^4n/16 \leq |S| \leq (1 - \rho^4/16)n$ with probability at least $1 - 1/n$ by Lemma 7.4. Hence by a union bound, we have $\rho^4n/16 \leq |S| \leq (1 - \rho^4/16)n$ with probability at least $1 - 2/n$.

Finally, we analyze the average sensitivity. Fix $\alpha > 0$. Let $A$ denote Algorithm 5 run with parameter $\alpha$. Fix $e \in E$. Let $f$ and $f_e$ denote the random variables corresponding to the output of the procedure \textsc{Embedding} when $A$ is run on graphs $G$ and $G - e$, respectively. Let $\lambda_0$ denote a joint distribution on the random variables $f$ and $f_e$. Similarly, let $d$ and $d_e$ denote the random variables resulting from executing Step 2 of $A$ graphs $G$ and $G - e$, respectively. Let $\lambda_1$ be a joint distribution on $d$ and $d_e$.

$$d_{\text{EM}}(A(G), A(G - e)) = d_{\text{EM}}(\text{\textsc{Rounding}}(f, \alpha), \text{\textsc{Rounding}}(f_e, \alpha))$$

$$= O\left(\frac{1}{\rho^2}\right) \int \|f - f_e\|_1 \lambda_0(f, f_e) d(f, f_e)$$

(by Lemma 7.4)
\[ O \left( \frac{1}{\rho^2} \right) \cdot d_{\text{EM}}^3(f, f_e) \]
\[ = O \left( \frac{\log^2 n}{\rho^2} \right) \int \|d - d_e\|_1 \lambda_1(d, d_e) d(d, d_e) \] 
(by Lemma 7.3)
\[ = O \left( \frac{\log^2 n}{\rho^2} \right) \cdot d_{\text{EM}}^3(d, d_e) = O \left( \frac{\log^4 n}{\rho^2} \right). \]

8 Maximum Matching

A vertex-disjoint set of edges is called a matching. In the maximum matching problem, given a graph, we want to find a matching of the maximum size. In this section, we describe several algorithms with low average sensitivity that approximate the maximum matching in a graph.

8.1 Lexicographically smallest matching

In this section, we describe an algorithm that computes a maximum matching in a graph with average sensitivity at most \( \frac{\text{OPT}^2}{m} \) and prove Theorem 8.1, where \( \text{OPT} \) is the maximum size of a matching.

First, we define some ordering among vertex pairs. Then, we can naturally define the lexicographical order among matchings by regarding a matching as a sorted sequence of vertex pairs. Then, our algorithm simply outputs the lexicographically smallest matching. Note that this can be done in polynomial time using Edmonds' algorithm \([8]\).

Theorem 8.1. Let \( A \) be the algorithm that outputs the lexicographically smallest maximum matching. Then, the average sensitivity of \( A \) is at most \( \frac{\text{OPT}^2}{m} \), where \( \text{OPT} \) is the maximum size of a matching.

Proof. For a graph \( G = (V, E) \), let \( M(G) \subseteq E \) be its lexicographically smallest maximum matching. As long as \( e \notin M \), we have \( M(G) = M(G - e) \). Hence, the average sensitivity of the algorithm is at most
\[ \frac{\text{OPT}}{m} \cdot \text{OPT} + \left( 1 - \frac{\text{OPT}}{m} \right) \cdot 0 = \frac{\text{OPT}^2}{m}. \]
\[ \square \]

Remark 8.2. Consider the path graph \( P_n = (\{1, \ldots, n\}, E) \), where \( E = \{(i, i + 1) : i \in [n - 1]\} \). The average sensitivity of the above algorithm on \( P_n \) is \( \Omega \left( \frac{\text{OPT}^2}{m} \right) \). Hence the above analysis of the average sensitivity is tight.

8.2 Greedy matching algorithm

In this section, we analyze the average sensitivity of the randomized greedy algorithm (Algorithm 6) that outputs a maximal matching. It is evident that Algorithm 6 runs in polynomial time and that the matching it outputs has size at least \( \frac{1}{2} \) the size of a maximum matching in the input graph.

Theorem 8.3. Algorithm 6 is a \( \frac{1}{2} \)-approximation algorithm for the maximum matching problem and has average sensitivity at most 1.
Algorithm 6: Randomized Greedy Algorithm

**Input:** undirected unweighted graph $G = (V, E)$

1. Sample a uniformly random ordering $\pi$ of edges in $E$;
2. Set $M \leftarrow \emptyset$;
3. Consider edges one by one according to $\pi$ and add an edge $(u, v)$ to $M$ only if both $u$ and $v$ are unmatched in $M$;
4. Return $M$.

Proof. For a permutation $\pi$ of edges in $E$, let $M_\pi(G)$ denote the matching obtained by running Algorithm 6 on a graph $G$. Using [6, Theorem 1], we get that for every $e \in E(G)$, it holds that $\mathbb{E}_\pi[\text{Ham}(M_\pi(G), M_\pi(G - e))] \leq 1$. This implies that the average sensitivity of Algorithm 6 is at most 1.

A vertex set $S \subseteq V$ in a graph $G = (V, E)$ is called a vertex cover if every edge in $E$ is incident to a vertex in $S$. In the minimum vertex cover problem, given a graph $G$, we want to compute a vertex cover of the minimum size. It is well known that, for any maximal matching $M$, the vertex set consisting of all endpoints of edges in $M$ is a 2-approximate vertex cover. The following theorem is immediate from Theorem 8.3.

**Theorem 8.4.** There exists a 2-approximation algorithm for the minimum vertex cover problem with average sensitivity at most 2.

### 8.3 Matching algorithm based on augmenting paths

In this section, we describe a $(1 - \varepsilon)$-approximation algorithm for the maximum matching problem with average sensitivity $\tilde{O}\left(\frac{\text{OPT}}{\varepsilon^{c+1}}/\varepsilon^{c+1}\right)$ for $c = O(1/\varepsilon^2)$ in Theorem 8.11. The basic building block is a $(1 - \varepsilon)$-approximation algorithm (Algorithm 7) for maximum matching that is based on iteratively augmenting a matching with greedily chosen augmenting paths of increasing lengths. In Theorem 8.5, we show that the average sensitivity of this algorithm is $\Delta^{O(1/\varepsilon^2)}$, where $\Delta$ is the maximum degree of the input graph. We obtain Theorem 8.5 by applying Theorem 1.9 to a result by Yoshida et al. [36].

We then apply Theorem 8.6 to Theorem 8.5 in order to get rid of the dependence of the average sensitivity on the maximum degree and obtain Theorem 8.10. We then combine (using Theorem 1.8, the parallel composition theorem) the algorithm guaranteed by Theorem 8.10 with the algorithm guaranteed by Theorem 8.1 to obtain Theorem 8.11.

#### 8.3.1 Greedy matching algorithm based on augmenting paths

In this section, we present an approximation algorithm that starts with an empty matching and then iteratively improves its size with augmenting paths of increasing lengths. We show that the average sensitivity of this algorithm can be bounded using Theorem 1.9.

**Theorem 8.5.** Algorithm 7 with parameter $\varepsilon > 0$ has approximation ratio $1 - \varepsilon$ and average sensitivity $\Delta^{O(1/\varepsilon^2)}$, where $\Delta$ is the maximum degree of the input graph.
Algorithm 7: Greedy Augmenting Paths Algorithm

Input: undirected unweighted graph \( G = (V, E) \), parameter \( \varepsilon \in (0, 1) \)

1. \( M_0 \leftarrow \emptyset \);
2. for \( i \in \{1, 2, \ldots , \lfloor \frac{1}{\varepsilon} - 1 \rfloor \} \) do
   3. Let \( A_i \) denote the set of augmenting paths of length \( 2i - 1 \) for the matching \( M_{i-1} \);
   4. Let \( A_i' \) denote a maximal set of disjoint paths from \( A_i \), where \( A_i' \) is made from a
      random ordering of \( A_i \);
   5. \( M_i \leftarrow M_{i-1} \triangle A_i' \);
6. return \( M_{\lfloor \frac{1}{\varepsilon} - 1 \rfloor} \).

Proof. For all \( k \geq 0 \), it is known that \( |M_k| \geq \frac{k}{k+1} \cdot |M^*| \) \(^{[12]}\), where \( M^* \) denotes a maximum

Yoshida et al. \(^{[36, Theorem 3.7]}\) show that for all \( k \geq 0 \), determining whether a uniformly

random edge \( e \sim E \) belongs to \( M_k \) can be done by querying at most \( \Delta O(k^2) \) edges in expectation,

where \( \Delta \) is the maximum degree of \( G \). Applying Theorem 1.9 to this, we can see that the average
sensitivity of Algorithm 7 with parameter \( \varepsilon > 0 \) and input \( G \) is \( \Delta O(1/\varepsilon^2) \), where \( \Delta \) is the maximum
degree of \( G \). \( \square \)

8.3.2 Stable-on-average thresholding transformation

In this section, we show a transformation from matching algorithms whose average sensitivity is a
function of the maximum degree to matching algorithms whose average sensitivity does not depend
on the maximum degree. This is done by adding to the algorithm, a preprocessing step that removes
vertices from the input graph, where the removed vertices have degree at least an appropriate
random threshold. Such a transformation helps us to design stable-on-average algorithms for graphs
with unbounded degree. Let \( \text{Lap}(\mu, \phi) \) denote the Laplace distribution with a location parameter
\( \mu \) and a scale parameter \( \phi \).

Theorem 8.6. Let \( A' \) be a randomized algorithm for the maximum matching problem such that
the size of the matching output by \( A' \) on a graph \( G \) is always at least \( a \cdot \text{OPT} \) for some \( a \geq 0 \). In
addition, assume that there exists an oracle \( O \) satisfying the following:

- when given access to a graph \( G = (V, E) \) and query \( e \in E \), the oracle generates a random
  string \( \pi \in \{0, 1\}^r \) and outputs whether \( e \) is contained in the matching output by \( A' \) on \( G \) with
  \( \pi \) as its random string, and
- the oracle \( O \) makes at most \( q(\Delta) \) queries to \( G \) in expectation, where \( \Delta \) is the maximum degree
  of \( G \) and the expectation is taken over the random coins of \( A' \) and a uniformly random query
  \( e \in E \).

Let \( \delta > 0 \) and \( \tau \) be a non-negative function on graphs. Then, there exists an algorithm \( A \) for the
maximum matching problem with average sensitivity

\[
\beta(G) \leq O \left( \frac{K_G}{\delta(\tau(G) - K_G)} + \exp \left( -\frac{1}{\delta} \right) \right) \cdot \text{OPT} + \mathbb{E}_L \left[ (2L - 2)^2 q(L) \right],
\]
where $L$ is a random variable distributed as $\text{Lap}(\tau(G), \delta \tau(G))$ and $K_G = \max_{e \in E(G)} |\tau(G) - \tau(G - e)|$. Moreover, the expected size of the matching output by $A$ is at least

$$a \cdot \text{OPT} - \frac{am}{(1 - \delta \ln(\text{OPT}/2)) \cdot \tau(G)} - a.$$

The following fact will be useful in the proof of Theorem 8.6.

Proposition 8.7. Let $L$ be a random variable distributed as $\text{Lap}(\mu, \phi)$. Then, $\Pr[L < (1 - \varepsilon)\mu] \leq \exp(-\varepsilon \mu/\phi)/2$. Similarly, $\Pr[L > (1 + \varepsilon)\mu] \leq \exp(-\varepsilon \mu/\phi)/2$.

Proof of Theorem 8.6. The algorithm $A$ is given below.

Algorithm $A$: On input $G = (V, E)$,

1. Sample a random variable $L$ according to the distribution $\text{Lap}(\tau(G), \delta \tau(G))$.
2. Let $[G]_L$ be the graph obtained after removing from $G$ all vertices of degree at least $L$.

We first bound the average sensitivity of $A$. We can think of $A$ as being sequentially composed of two algorithms, where the first algorithm takes in a graph $G = (V, E)$ and outputs a number $L \sim \text{Lap}(\tau(G), \delta \tau(G))$. The second algorithm takes both $L$ and $G$ and runs $A'$ on $[G]_L$. Let $L_e$ for $e \in E$ denote a Laplace random variable distributed as $\text{Lap}(\tau(G - e), \delta \tau(G - e))$. Using Theorem 1.6 we get that the average sensitivity of $A$ is bounded by

$$\text{OPT} \cdot \mathbb{E}_{e \sim \text{TV}}[d_{TV}(L, L_e)] + \mathbb{E}_L \left[ \mathbb{E}_{e \sim \text{EM}}[d_{\text{EM}}(A'([G]_L), A'([G - e]_L))] \right].$$

Claim 8.8. For $x \in \mathbb{R}$, $\mathbb{E}_{e \sim \text{EM}}[d_{\text{EM}}(A'([G]_x), A'([G - e]_x))] \leq (2x - 2)^2 q(x)$.

Proof. Fix $x \in \mathbb{R}$. In order to bound the term $\mathbb{E}_{e \sim \text{EM}}[d_{\text{EM}}(A'([G]_x), A'([G - e]_x))]$, consider the following algorithm $A'_x$. On input $G = (V, E)$, the algorithm $A'_x$ first removes every vertex of degree at least $x$ from $G$ and then runs $A'$ on the resulting graph. Hence, the quantity $\mathbb{E}_{e \sim \text{EM}}[d_{\text{EM}}(A'([G]_x), A'([G - e]_x))]$ denotes the average sensitivity of $A'_x$.

In order to bound the average sensitivity of $A'_x$, construct an oracle $O_x$ as follows. $O_x$ when given access to a graph $G = (V, E)$ and input $e$ sampled uniformly at random from $E$, does the following. It first checks whether at least one of the endpoints of $e$ has degree at least $x$. If so, it returns that $e$ does not belong to the solution obtained by running $A'_x$ on $G$. Otherwise, it runs $O$ with access to $[G]_x$ and $e$ as input and outputs the answer of $O$.

We can analyze the query complexity of $O_x$ as follows. Call an edge $e \in E$ alive if both the endpoints of $e$ have degree less than $x$. Otherwise, $e$ is dead.

The oracle $O_x$ can check whether an edge $e = (u, v)$ is alive or not by querying at most $2x - 2$ edges incident to $e$. In particular $O_x$ examines the neighbors of $u$ and $v$ one by one, and, as soon $O_x$ encounters $x - 1$ distinct neighbors (excluding $u$ or $v$ themselves) for either $u$ or $v$, $O_x$ can declare $e$ to be a dead edge.

If the edge $e \in E$ input to $O_x$ is a dead edge, therefore, $O_x$ queries at most $2x - 2$ edges and returns that $e$ cannot be part of a solution to running $A'_x$ on $G$.  

32
If the input edge \( e \in E \) is alive, then we know that it is a uniformly random alive edge. By the guarantee on \( O \), we then know that \( O \) makes at most \( q(x) \) queries to the alive edges in expectation over the randomness of \( A' \) and the choice of the input alive edge, since the maximum degree of \( [G]_x \) is at most \( x \). In order for the oracle \( O_x \) to simulate oracle access to \( [G]_x \) for the oracle \( O \), for each alive edge \( e \) queried by \( O \), the oracle \( O_x \) has to query each edge incident to \( e \) in \( G \) and determine which among these are alive. Since \( e \) is alive, both endpoints of \( e \) have degrees less than \( x \). Hence, \( O_x \) need only check whether at most \( 2x - 2 \) edges incident to \( e \) are alive or not. This can be done by querying \( (2x - 2)^2 \) edges in \( E \) in total.

Combining all of the above, the expected query complexity of \( O_x \) is at most \( (2x - 2)^2 q(x) \), where the expectation is taken over the edges of \( e \in E \) and the randomness in \( A_x \).

Therefore, by Theorem 1.9, we get that the average sensitivity of algorithm \( A_x \) is bounded by \( (2x - 2)^2 q(x) \).

We now bound the quantity \( \mathbb{E}_{e \sim E} [d_{TV}(L, L_e)] \).

**Claim 8.9.** For any \( e \in E \), we have

\[
d_{TV}(L, L_e) \leq O \left( \frac{K}{\delta(\tau - K)} + \exp \left( \frac{-1}{\delta} \right) \right).
\]

**Proof.** Let \( f_L, f_{L_e} : \mathbb{R} \to \mathbb{R} \) be the probability density functions of the Laplace random variables \( L \) and \( L_e \), respectively. Let \( \tau = \tau(G) \), \( \tau_e = \tau(G_e) \), and \( K = K_G \). Then

\[
\frac{f_L(x)}{f_{L_e}(x)} = \frac{1}{2\delta} \exp \left( -\frac{|x - \tau|}{\delta \tau} \right) = \frac{\tau_e}{\tau} \exp \left( \frac{|x - \tau_e|}{\delta \tau_e} - \frac{|x - \tau|}{\delta \tau} \right) = \left( 1 - \frac{\tau - \tau_e}{\tau} \right) \exp \left( \frac{\tau |x - \tau_e| - \tau_e |x - \tau|}{\delta \tau \tau_e} \right).
\]

A direct calculation shows that for \( 0 \leq x \leq 2 \max \{ \tau, \tau_e \} \), we have

\[
\left( 1 - \frac{K}{\tau} \right) \exp \left( \frac{-2K}{\delta(\tau - K)} \right) \leq \frac{f_L(x)}{f_{L_e}(x)} \leq \left( 1 + \frac{K}{\tau} \right) \exp \left( \frac{2K}{\delta(\tau - K)} \right).
\]

This implies that for all \( S \subseteq [0, 2 \max \{ \tau, \tau_e \}] \),

\[
\left( 1 - \frac{K}{\tau} \right) \exp \left( \frac{-2K}{\delta(\tau - K)} \right) - 1 \leq \Pr[L \in S] - \Pr[L_e \in S] \leq \left( 1 + \frac{K}{\tau} \right) \exp \left( \frac{2K}{\delta(\tau - K)} \right) - 1.
\]

By Proposition 8.7, the probability that \( L \) (and \( L_e \) as well) falls in the range \([-\infty, 0] \cup [2 \max \{ \tau, \tau_e \}, \infty] \) is bounded by \( \exp(1/\delta) \). Hence, total variation distance between \( L \) and \( L_e \) is

\[
\left( 1 + \frac{K}{\tau} \right) \exp \left( \frac{2K}{\delta(\tau - K)} \right) - \left( 1 - \frac{K}{\tau} \right) \exp \left( \frac{-2K}{\delta(\tau - K)} \right) + 2 \exp \left( \frac{-1}{\delta} \right)
\]

\[
= \left( 1 + \frac{K}{\tau} \right) \left( 1 + \frac{2K}{\delta(\tau - K)} + O \left( \frac{K^2}{\delta^2(\tau - K)^2} \right) \right) - \left( 1 - \frac{K}{\tau} \right) \left( 1 - \frac{2K}{\delta(\tau - K)} - O \left( \frac{K^2}{\delta^2(\tau - K)^2} \right) \right) + 2 \exp \left( \frac{-1}{\delta} \right)
\]

33
Therefore, the average sensitivity of $A$ is bounded as
\[
\beta(G) = \mathbb{E}_{e \sim E} d_{\text{EM}}(A(G), A(G - e)) \\
\leq O\left(\frac{K}{\delta(\tau - K)} + \exp\left(-\frac{1}{\delta}\right)\right) \cdot \text{OPT} + \mathbb{E}_{x} \left[(2x - 2)^2 q(x)\right].
\]

We now bound the approximation guarantee of $A$. By Proposition 8.7,
\[
\Pr\left[L < \left(1 - \delta \ln\left(\frac{\text{OPT}}{2}\right)\right) \cdot \tau(G)\right] \leq \frac{1}{\text{OPT}}.
\]

Therefore, with probability at least $1 - 1/\text{OPT}$, only those vertices with degree at least $(1 - \delta \ln(\text{OPT}/2)) \cdot \tau(G)$ are removed from $G$. The number of such vertices is at most $\frac{m}{(1 - \delta \ln(\text{OPT}/2)) \cdot \tau(G)}$.

Therefore, with probability at least $1 - 1/\text{OPT}$, the size of a maximum matching in the resulting graph is at most $\frac{m}{(1 - \delta \ln(\text{OPT}(G)/2)) \cdot \tau(G)}$ smaller than that of $G$. With probability at most $1/\text{OPT}$, the size of a maximum matching in the resulting instance could be smaller by an additive term of at most $\text{OPT}$. Hence, the expected size of a maximum matching in the new instance is at least
\[
\text{OPT} - \frac{m}{(1 - \delta \ln(\text{OPT}(G)/2)) \cdot \tau(G)} - 1.
\]

The statement on approximation guarantee follows.

---

### 8.3.3 Average sensitivity of the greedy augmenting paths algorithm with thresholding

**Theorem 8.10.** Let $\varepsilon \in (0, 1)$ be a parameter. There exists an algorithm with approximation ratio $1 - \varepsilon$ and average sensitivity
\[
O\left(\frac{\varepsilon}{1 - \varepsilon} \log n\right) + \left(\frac{m}{\varepsilon^3 \text{OPT}}\right)^{O(1/\varepsilon^2)}.
\]

**Proof.** The algorithm guaranteed by the theorem statement is as follows.

**Algorithm $A_\varepsilon$:** On input $G = (V, E)$,

1. Compute $\text{OPT}$.

2. If $\text{OPT} \leq \frac{2}{\varepsilon} + 1$ or $m \leq \frac{1}{3\varepsilon}$, then output an arbitrary maximum matching.

3. Otherwise, run the algorithm obtained by applying Theorem 8.6 with the setting $\tau := \tau(G) = \frac{m}{\varepsilon \text{OPT}}$ and $\delta := \frac{1}{2 \ln n}$ to Algorithm 7 run with parameter $\varepsilon'$, where $\varepsilon' = \frac{\varepsilon}{3} - \frac{1}{3\text{OPT}}$.

34
**Approximation guarantee:** If $\OPT \leq \frac{1}{2} + 1$ or $m \leq \frac{1}{2\varepsilon}$, the approximation guarantee is clear. Otherwise, since Algorithm 7 outputs a maximal matching whose size is always at least $(1 - \varepsilon') \cdot \OPT$, the size of the matching output by $A_e$ is at least $(1 - \varepsilon') \cdot \OPT - \varepsilon'(1 - \varepsilon') \cdot \OPT = \frac{\OPT}{1 - \ln(\OPT/2\varepsilon)} - (1 - \varepsilon')$, which is at least $(1 - \varepsilon) \cdot \OPT$ by the setting of $\varepsilon'$ and the fact that $\frac{\ln(\OPT/2\varepsilon)}{2\ln n} \leq \frac{1}{2}$.

**Average sensitivity:** If $\OPT \leq \frac{2}{\varepsilon} + 1$ or $m \leq \frac{1}{2\varepsilon}$, the average sensitivity of $A_e$ is bounded by $O(1)$, since the size of maximum matching in $G$ is small and it can decrease only by at most 1 by the removal of an edge.

We now analyze the average sensitivity of $A_e$ for the case that $\OPT > \frac{2}{\varepsilon} + 1$ and $m > \frac{1}{2\varepsilon}$. Let $c = O(1/\varepsilon^2)$. The average sensitivity of the algorithm resulting from applying Theorem 8.6 to Algorithm 7 is bounded as:

$$O\left(\frac{K_G}{\delta(\tau - K_G)} + \exp\left(-\frac{1}{\delta}\right)\right) \cdot \OPT + \int_0^\infty (2x - 2)^2 \cdot x^c \cdot \frac{1}{2\delta \tau} \cdot \exp\left(-\frac{|x - \tau|}{\delta \tau}\right) \ dx. \quad (13)$$

To obtain the above expression, we used the fact (from [36, Theorem 3.7]) that $q(x) \leq x^c$ when $x > 0$ and $q(x) = 0$ otherwise.

The second term of (13) can be bounded as:

$$\int_0^\infty (2x - 2)^2 \cdot x^c \cdot \frac{1}{2\delta \tau} \cdot \exp\left(-\frac{|x - \tau|}{\delta \tau}\right) \ dx = 4 \int_\tau^{\infty} x^{c+2} \cdot \frac{1}{\delta \tau} \cdot \exp\left(-\frac{x - \tau}{\delta \tau}\right) \ dx$$

$$= \exp\left(\frac{1}{\delta}\right) (\delta \tau)^{c+2} \Gamma(c + 3, \frac{1}{\delta}) = (\delta \tau)^{c+2} (c + 2)! \sum_{k=0}^{c+2} \frac{(1/\delta)^k}{k!} = \left(\frac{m}{\varepsilon^2 \OPT}\right)^{O(1/\varepsilon^2)}$$

where $\Gamma(s, \cdot)$ is the incomplete Gamma function and we have used the fact that $\Gamma(s + 1, x) = s! \exp(-x) \sum_{k=0}^{s} x^k / k!$ if $s$ is a non-negative integer. Moreover, each term in the summation $\delta^{c+2}$

$(c + 2)! \sum_{k=0}^{c+2} \frac{(1/\delta)^k}{k!}$ is $o(1)$. Hence, the summation is $O(\frac{1}{\varepsilon^2})$.

In order to bound the first term of (13), note that

$$K_G = \max_{e \in E} \left| \tau(G) - \tau(G - e) \right|$$

$$= 3 \max_{e \in E} \left| \frac{m}{\varepsilon \OPT(G) - 1} - \frac{m - 1}{\varepsilon \OPT(G - e) - 1} \right|$$

$$\leq 3 \max \left\{ \frac{m}{\varepsilon \OPT(G) - 1} - \frac{m - 1}{\varepsilon \OPT(G) - 1}, \frac{m - 1}{\varepsilon \OPT(G) - 1} - \frac{m}{\varepsilon \OPT(G) - 1} \right\}$$

$$= 3 \max \left\{ \frac{1}{\varepsilon \OPT(G) - 1}, \frac{m - \OPT(G)}{\varepsilon \OPT(G) - 1} \cdot (\varepsilon \OPT(G)) \right\}$$

$$= \frac{3}{\varepsilon \OPT(G) - 1} \max \left\{ 1, \frac{m - \OPT(G) + 1}{\varepsilon \OPT(G) - 1} \right\}. $$

The inequality above uses the fact that for numbers $u, v, w \geq 0$ such that $u \leq v$ and $w(v-1) - 1 \geq 0$, we have that $\frac{u}{wv - 1} \leq \frac{u - 1}{w(v-1) - 1}$.

Since $\frac{\varepsilon(m - \OPT(G) + 1)}{\varepsilon \OPT(G) - 1}$ is a nonincreasing function of $\OPT$ and $\OPT > \frac{2}{\varepsilon} + 1$, we have that

$$\frac{\varepsilon(m - \OPT(G) + 1)}{\varepsilon \OPT(G) - 1} < \varepsilon m.$$

35
Hence, $K_G < \frac{3}{\OPT - 1} \max \{1, \varepsilon m\} = \frac{9\varepsilon m}{\OPT - 1}$, since $m > \frac{1}{3\varepsilon}$ and therefore, we have that $\tau - K_G \geq \tau(1 - 9\varepsilon)$. Hence, the first term of (13) can be upper bounded by

$$O\left(\frac{K_G}{\delta \tau(1 - 9\varepsilon)} + \exp\left(-\frac{1}{\delta}\right)\right) \cdot \OPT = O\left(\frac{9\varepsilon m}{\varepsilon \OPT - 1} \cdot \frac{1}{1 - 9\varepsilon} \cdot \frac{2\ln n}{m^{\OPT - 1}} + \OPT \right)$$

$$= O\left(\frac{\varepsilon}{1 - \varepsilon} \cdot \log n\right).$$

Hence, the average sensitivity of the algorithm obtained can be bounded by:

$$\beta(G) = \max\left\{O\left(\frac{1}{\varepsilon}\right), O\left(\frac{\varepsilon}{1 - \varepsilon} \cdot \log n\right) + \left(\frac{m}{\varepsilon^3 \OPT}\right)^{O(1/\varepsilon^2)}\right\}$$

$$= O\left(\frac{\varepsilon}{1 - \varepsilon} \cdot \log n\right) + \left(\frac{m}{\varepsilon^3 \OPT}\right)^{O(1/\varepsilon^2)}.$$ 

\[\square\]

### 8.3.4 Average sensitivity of a combined matching algorithm

In this section, we combine the algorithms guaranteed by Theorems 8.1 and 8.10 in order to get a matching algorithm with improved sensitivity.

**Theorem 8.11.** Let $\varepsilon \in (0, 1)$ be a parameter. There exists an algorithm with approximation ratio $1 - \varepsilon$ and average sensitivity

$$\OPT(G) \cdot \frac{\varepsilon}{1 - \varepsilon} \cdot O\left(\left(\frac{\varepsilon}{1 - \varepsilon} \cdot \log n\right)^{\frac{1}{c+1}} + \frac{1}{\varepsilon^{c+1}}\right)$$

for $c = O(1/\varepsilon^2)$.

**Proof.** Let $c = O(1/\varepsilon^2)$. The algorithm guaranteed by the theorem is given as Algorithm 8. The bounds on approximation guarantee and average sensitivity are both straightforward when $\OPT < 2c$ or $m < 2c$.

**Algorithm 8:** COMBINED ALGORITHM TO $(1 - \varepsilon)$-APPROXIMATE maximum MATCHING

**Input:** undirected unweighted graph $G = (V, E)$

1. Compute $\OPT$;
2. if $\OPT < 2c$ or $m < 2c$ then
   3. return an arbitrary maximum matching in $G$.
4. else
   5. Let $f(G) \leftarrow \frac{\OPT^2}{m}$ and $g(G) \leftarrow \frac{\varepsilon}{1 - \varepsilon} \cdot \log n + \left(\frac{m}{\varepsilon^3 \OPT}\right)^c$;
6. Run the algorithm given by Theorem 8.1 with probability $\frac{g(G)}{f(G) + g(G)}$ and run the algorithm given by Theorem 8.10 with the remaining probability.

The approximation guarantee in the case when $\OPT \geq 2c$ and $m \geq 2c$ is also straightforward since Algorithm 8 is simply a distribution over algorithms guaranteed by Theorem 8.1 and Theorem 8.10.
We now bound the average sensitivity of Algorithm 8.1 when \( \text{OPT} \geq 2c \) and \( m \geq 2c \). Let \( \rho(G) \) denote the probability \( \frac{g(G)}{f(G)+g(G)} \). By Theorem 10.2, the average sensitivity is at most

\[
\frac{O(f(G)) \cdot g(G) + O(g(G)) \cdot f(G)}{f(G) + g(G)} + 2\text{OPT} \cdot \mathbb{E}_{e \sim E} [\rho(G) - \rho(G-e)].
\]  

(14)

We first bound the quantity \( \mathbb{E}_{e \sim E} [\rho(G) - \rho(G-e)] \).

**Claim 8.12.** For every graph \( G = (V, E) \) such that \( \text{OPT} \geq c + 1 \), and for every \( e \in E \),

\[
\left(1 - \frac{c}{m}\right) \cdot g(G) \leq g(G-e) \leq \left(1 + \frac{c}{\text{OPT} - c}\right) \cdot g(G).
\]

**Proof.** We first prove the upper bound. We know that

\[
g(G-e) \leq \left(1 + \left(\frac{m-1}{\varepsilon \log n + \frac{m^c}{\varepsilon^c \text{OPT}}}\right)^c - \left(\frac{m}{\varepsilon^c \text{OPT}}\right)^c\right) \leq \left(1 + \left(\frac{m-1}{\varepsilon \log n + \frac{m^c}{\varepsilon^c \text{OPT}}}\right)^c\right).
\]

Note that the last inequality holds whenever \( \text{OPT} > c \), because \((1+x)^r \leq 1 + \frac{rx}{1-(r-1)x}\) for \( x \in [0, \frac{1}{r-1}] \) and \( r > 1 \).

For the lower bound,

\[
g(G-e) \geq \left(1 - \left(\frac{m}{\varepsilon^c \text{OPT}}\right)^c - \left(\frac{m-1}{\varepsilon \log n + \frac{m^c}{\varepsilon^c \text{OPT}}}\right)^c\right) \geq \left(1 - \left(\frac{m}{\varepsilon^c \text{OPT}}\right)^c - \left(\frac{m-1}{\varepsilon \log n + \frac{m^c}{\varepsilon^c \text{OPT}}}\right)^c\right)
\]

\[
= \left(1 - \frac{1}{m}\right)^c \geq 1 - \frac{c}{m}.
\]

**Claim 8.13.** For every graph \( G = (V, E) \) and every \( e \in E \),

\[
f(G) \cdot \left(1 - \frac{2}{\text{OPT}}\right) \leq f(G-e) \leq f(G) \cdot \left(1 + \frac{1}{m-1}\right).
\]

**Proof.** To prove the upper bound,

\[
\frac{f(G-e)}{f(G)} \leq \left(\frac{m}{m-1}\right) = \left(1 + \frac{1}{m-1}\right).
\]

For the lower bound,

\[
\frac{f(G-e)}{f(G)} \geq \left(\frac{\text{OPT}-1}{\text{OPT}}\right)^2 \cdot \left(\frac{m}{m-1}\right)^2 \geq \left(\frac{\text{OPT}-1}{\text{OPT}}\right)^2 \geq 1 - \frac{2}{\text{OPT}}.
\]

**Claim 8.14.** For every graph \( G = (V, E) \) such that \( \text{OPT} \geq 2c \) and \( m \geq 2c \), and for every \( e \in E \),

\[
\rho(G) \cdot \left(1 - \frac{2c}{\text{OPT}-c}\right) \leq \rho(G-e) \leq \rho(G) \cdot \left(1 + \frac{5c}{\text{OPT}-c}\right).
\]
Proof. Note that \((1 - \frac{2}{\text{OPT}})^{-1} \leq 1 + \frac{1}{\text{OPT}}\) and \((1 - \frac{c}{m})^{-1} \leq 1 + \frac{2}{m}\) for \(\text{OPT} \geq 4\) and \(m \geq 2c\). We also have \(\left(1 + \frac{c}{\text{OPT} - c}\right)^{-1} \geq 1 - \frac{c}{\text{OPT} - c}\) and \((1 + \frac{1}{m - 1})^{-1} \geq 1 - \frac{1}{m - 1}\) for \(\text{OPT} \geq 2c\) and \(m \geq 2\).

Combining all of the above,

\[
\rho(G - e) = \frac{g(G - e)}{f(G - e) + g(G - e)} \leq \frac{g(G) \cdot \left(1 + \frac{c}{\text{OPT} - c}\right)}{(f(G) + g(G)) \cdot \min \{1 - \frac{c}{m}, 1 - \frac{2}{\text{OPT}}\}} \leq \rho(G) \cdot \left(1 + \frac{c}{\text{OPT} - c}\right) \cdot \max \left\{1 + \frac{2c}{m}, 1 + \frac{4}{\text{OPT}}\right\} \leq \rho(G) \cdot \left(1 + \frac{5c}{\text{OPT} - c}\right). \quad \square
\]

Using similar calculations, we can see that

\[
\rho(G - e) \geq \rho(G) \cdot \left(1 - \frac{c}{m}\right) \cdot \min \left\{1 - \frac{c}{\text{OPT} - c}, 1 - \frac{1}{m - 1}\right\} \geq \rho(G) \cdot \left(1 - \frac{2c}{\text{OPT} - c}\right).
\]

Thus, for all \(e \in E\), we have that \(|\rho(G) - \rho(G - e)| \leq \max \left\{\frac{2c}{\text{OPT} - c}, \frac{5c}{\text{OPT} - c}\right\} \cdot \rho(G) = \frac{5c\rho(G)}{\text{OPT} - c} \cdot \rho(G)\) for \(\text{OPT} \geq 2c\).

Hence, \(\mathbb{E}_{e \sim E}[|\rho(G) - \rho(G - e)|] \leq \frac{5c\rho(G)}{\text{OPT} - c}\).

Therefore, the average sensitivity of Algorithm 8 is at most

\[
\frac{O(f(G)) \cdot g(G) + O(g(G)) \cdot f(G)}{f(G) + g(G)} + 2\text{OPT} \cdot \mathbb{E}_{e \sim E}[|\rho(G) - \rho(G - e)|]
\]

\[
= O\left(\frac{f(G)^{c/(c+1)} g(G)^{1/(c+1)}}{f(G)^{1/(c+1)} + g(G)^{c/(c+1)}}\right) + O\left(\frac{\text{OPT}^{\text{cp}(G)}}{\text{OPT}}\right)
\]

\[
= O\left(f(G)^{c/(c+1)} g(G)^{1/(c+1)}\right) + O(1/\varepsilon^2)
\]

\[
= O\left(\frac{\text{OPT}^2}{m}\right)^{c/(c+1)} \cdot \left(\frac{\varepsilon}{1 - \varepsilon}\right)^{1/(c+1)} \left(\log n\right)^{1/(c+1)} + O(1/\varepsilon^2)
\]

\[
= O\left(\frac{\text{OPT}^{2c/(c+1)}}{m^{c/(c+1)}} \left(\frac{\varepsilon}{1 - \varepsilon}\right)^{1/(c+1)} \log^{1/(c+1)} n + \frac{\text{OPT}^{2c/(c+1)}}{m^{c/(c+1)} \varepsilon^{3c/(c+1)} \text{OPT}^{c/(c+1)}}\right) + O(1/\varepsilon^2)
\]

\[
= O\left(\text{OPT}^{c/(c+1)} \left(\frac{\varepsilon}{1 - \varepsilon}\right)^{1/(c+1)} \log^{1/(c+1)} n + \frac{\text{OPT}^{c/(c+1)}}{\varepsilon^{3c/(c+1)}}\right) + O(1/\varepsilon^2)
\]

\[
= O\left(\text{OPT}^{c/(c+1)} \left(\frac{\varepsilon}{1 - \varepsilon}\right)^{1/(c+1)} \log^{1/(c+1)} n + \frac{1}{\varepsilon^{3c/(c+1)}}\right).
\]

To obtain the first term of the expression resulting from the first equality, we divide both the numerator and denominator by \(f(G)^{\frac{c}{c+1}} \cdot g(G)^{\frac{1}{c+1}}\). The second term of the first equality above follows since \(\frac{\text{OPT}}{\text{OPT} - c} \leq 2\) as \(\text{OPT} \geq 2c\).
8.4 Lower bound

In this section, we show a lower bound of $\Omega(n)$ for the problem of exactly computing the maximum matching in a graph.

**Theorem 8.15.** Every algorithm that exactly computes the maximum matching in a graph has average sensitivity $\Omega(n)$.

**Proof.** Let $n \in \mathbb{N}$ be even. Consider the cycle $C_n$ on $n$ vertices. $C_n$ has exactly two maximum matchings $M_1$ and $M_2$ of size $n/2$ each. Both $M_1$ and $M_2$ consist of alternating edges of the cycle. Let $A$ be an algorithm that outputs $M_1$ with probability $p$ and $M_2$ with probability $1 - p$. Assume, without loss of generality, that $p \geq \frac{1}{2}$. For every edge $e \in M_1$, the unique maximum matching in the odd-length path $G - e$ has Hamming distance $n - 1$ from $M_1$. Thus, for each $e \in M_1$, the earth mover’s distance between $A(G)$ and $A(G - e)$ is at least $\frac{n-1}{2}$. Hence, the average sensitivity of $A$ is at least $\frac{1}{n} \sum_{e \in M_1} \frac{n-1}{2} = \Omega(n)$. \qed

9 2-Coloring

In the 2-coloring problem, given a bipartite graph $G = (V, E)$, we are to output a (proper) 2-coloring on $G$, that is, an assignment $f : V \rightarrow \{0, 1\}$ such that $f(u) \neq f(v)$ for every edge $(u, v) \in E$. Clearly this problem can be solved in linear time. In this section, however, we show that there is no stable-on-average algorithm for the 2-coloring problem.

**Theorem 9.1.** Any (randomized) algorithm for the 2-coloring problem has average sensitivity $\Omega(n)$.

**Proof.** Suppose that there is a (randomized) algorithm $A$ whose average sensitivity is at most $\beta n$ for $\beta < 1/256$. In what follows, we assume that $n$, that is, the number of vertices in the input graph, is a multiple of 16.

Let $\mathcal{P}_n$ be the family of all possible paths on $n$ vertices, and let $\mathcal{Q}_n$ be the family of all possible graphs on $n$ vertices consisting of two paths. Note that $|\mathcal{P}_n| = n!/2$ and $|\mathcal{Q}_n| = (n - 1)n!/4$. Consider a bipartite graph $H = (\mathcal{P}_n, \mathcal{Q}_n; E)$, where a pair $(P, Q)$ is in $E$ if and only if $Q$ can be obtained by removing an edge in $P$. Note that each $P \in \mathcal{P}_n$ has $n - 1$ neighbors in $H$ and each $Q \in \mathcal{Q}_n$ has four neighbors in $H$.

We say that an edge $(P, Q) \in E$ is intimate if $d_{EM}(A(P), A(Q)) \leq 8\beta n$. We observe that for every $P \in \mathcal{P}_n$, at least a $7/8$-fraction of the edges incident to $P$ are intimate; otherwise

$$\mathbb{E}_{e \sim E(P)} \left[ d_{EM}(A(P), A(P - e)) \right] > \frac{1}{8} \cdot 8\beta n = \beta n,$$

which is a contradiction, where $E(P)$ denotes the set of edges in $P$.

We say that a graph $Q \in \mathcal{Q}_n$ is heavy if both components of $Q$ have at least $n/16$ vertices, and say that an edge $(P, Q) \in E$ is heavy if $Q$ is heavy. We observe that for every $P \in \mathcal{P}_n$, at least a $7/8$-fraction of the edges incident to $P$ are heavy.

We say that an edge $(P, Q) \in E$ is good if it is intimate and heavy. Observe that for every $P \in \mathcal{P}_n$, by the union bound, at least a $3/4$-fraction of the edges incident to $P$ are good. In particular, this means that the fraction of good edges in $H$ is at least $3/4$. Hence, there exists $Q^* \in \mathcal{Q}_n$ that has at least three good incident edges; otherwise the fraction of good edges in $H$ is at most $2/4 = 1/2$, which is a contradiction.

39
Let \( f_1, \ldots, f_4 \) be the four 2-colorings of \( Q^* \). As \( Q^* \) has three good incident edges, without loss of generality, there are adjacent paths \( P_1, P_2 \in \mathcal{P}_n \) such that both \((P_1, Q^*)\) and \((P_2, Q^*)\) are good, and there is no assignment that is a 2-coloring for both \( P_1 \) and \( P_2 \). Without loss of generality, we assume that \( f_1, f_2 \) are 2-colorings of \( P_1 \), and \( f_3, f_4 \) are 2-colorings of \( P_2 \). Note that \( d_{\text{Ham}}(f_i, f_j) \geq n/16 \) for \( i \neq j \) because \( Q \) is heavy. Let \( q_i = \Pr[A(Q^*) = f_i] \) for \( i \in [4] \). As the edge \((P_1, Q^*)\) is intimate, we have

\[
8\beta \geq d_{\text{EM}}(A(P_1), A(Q^*)) \geq \frac{n}{16} \left( |\Pr[A(P_1) = f_1] - q_1| + |\Pr[A(P_1) = f_2] - q_2| + q_3 + q_4 \right)
\]

and hence we must have \( q_1 + q_2 \geq 1 - 128\beta \). Considering \( d_{\text{EM}}(A(P_2), A(Q^*)) \), we also have \( q_3 + q_4 \geq 1 - 128\beta \). However,

\[
1 = q_1 + q_2 + q_3 + q_4 \geq (1 - 128\beta) + (1 - 128\beta) = 2 - 256\beta > 1
\]

as \( \beta < 1/256 \), which is a contradiction. \(\square\)

10 General Results on Average Sensitivity

In this section, we state and prove some basic properties of average sensitivity and show that locality guarantees of solutions output by an algorithm imply low average sensitivity for that algorithm.

10.1 Bounds on \( k \)-average sensitivity from bounds on average sensitivity

In this section, we prove Theorem 1.5 which says that, if an algorithm is stable-on-average against deleting a single edge, it is also stable-on-average against deleting multiple edges. We restate the theorem here.

**Theorem 1.5.** Let \( A \) be an algorithm for a graph problem with the average sensitivity given by \( f(n, m) \). Then, for any integer \( k \geq 1 \), the algorithm \( A \) has \( k \)-average sensitivity at most \( \sum_{i=1}^{k} f(n, m - i + 1) \).

**Proof.** We have

\[
\mathbb{E}_{(e_1, \ldots, e_k) \sim (E_k)} \left[ d_{\text{EM}}(A(G), A(G - \{e_1, \ldots, e_k\})) \right]
\]

\[
\leq \mathbb{E}_{(e_1, \ldots, e_k) \sim (E_k)} \left[ \sum_{i=1}^{k} d_{\text{EM}}(A(G - \{e_1, \ldots, e_{i-1}\}), A(G - \{e_1, \ldots, e_i\})) \right]
\]

\[
= \mathbb{E}_{e_1 \sim E} \left[ d_{\text{EM}}(A(G), A(G - \{e_1\})) \right] + \mathbb{E}_{e_2 \sim E \setminus \{e_1\}} \left[ d_{\text{EM}}(A(G - \{e_1\}), A(G - \{e_1, e_2\})) \right] + \ldots
\]

\[
+ \mathbb{E}_{e_k \sim E \setminus \{e_1, \ldots, e_{k-1}\}} \left[ d_{\text{EM}}(A(G - \{e_1, \ldots, e_{k-1}\}), A(G - \{e_1, \ldots, e_k\})) \right] \ldots \]

\[
= f(n, m) + \mathbb{E}_{e_1 \sim E} \left[ \beta(G - \{e_1\}) \right] + \mathbb{E}_{e_2 \sim E \setminus \{e_1\}} \left[ \beta(G - \{e_1, e_2\}) \right] + \ldots
\]

\[
+ \mathbb{E}_{e_{k-1} \sim E \setminus \{e_1, \ldots, e_{k-2}\}} \left[ \beta(G - \{e_1, \ldots, e_{k-1}\}) \right] \ldots \]

40
We are able to interchange the order of expectations because of Fubini’s theorem \[11\].

\[\sum_{i=1}^{k} f(n, m - i + 1).\]

Here, the first inequality is due to the triangle inequality.

10.2 Sequential composition

In this section, we state and prove our two sequential composition theorems Theorem 1.6 and Theorem 1.7.

**Theorem 1.6** (Sequential composition). Consider two randomized algorithms \( A_1 : G \to S_1, A_2 : G \times S_1 \to S_2 \). Suppose that the average sensitivity of \( A_1 \) with respect to the total variation distance is \( \gamma_1 \) and the average sensitivity of \( A_2(\cdot, S_1) \) is \( \beta_2(S_1) \) for any \( S_1 \in S_1 \). Let \( A \) be a randomized algorithm obtained by composing \( A_1 \) and \( A_2 \), that is, \( A(G) = A_2(G, A_1(G)) \). Then, the average sensitivity of \( A \) is \( H \cdot \gamma_1(G) + \mathbb{E}_{S_1 \sim A_1(G)} \beta_2(S_1)(G) \), where \( H \) denotes the maximum Hamming weight among those of solutions obtained by running \( A \) on \( G \) and \( \{G - e\} \) over all \( e \in E \).

**Proof.** Consider \( G = (V, E) \) and let \( e \in E \). We bound the earth mover’s distance between \( A(G) \) and \( A(G - e) \) as follows. For a distribution \( D \), we use \( f_D \) to denote its probability mass function. We know that for all \( S_1 \in S_1 \) and \( S_2 \in S_2 \)

\[ f_{(A_1(G), A_2(G, S_1))}(S_1, S_2) = f_{A_1(G)}(S_1) \cdot f_{A_2(G, S_1)}(S_2), \]

where \( (A_1(G), A_2(G, S_1)) \) denotes the joint distribution of \( A_1(G) \) and \( A_2(G, S_1) \). Fix \( S_1 \in S_1 \). For each \( S_2 \in S_2 \), we transform probabilities of the form \( f_{(A_1(G), A_2(G, S_1))}(S_1, S_2) \) to \( f_{A_1(G)}(S_1) \cdot f_{A_2(G - e, S_1)}(S_2) \). This incurs a total cost of \( f_{A_1(G)}(S_1) \cdot d_{EM}(A_2(G, S_1), A_2(G - e, S_1)) \). We can now, for each \( S_1 \in S_1 \) and \( S_2 \in S_2 \), transform the probability \( f_{A_1(G)}(S_1) \cdot f_{A_2(G - e, S_1)}(S_2) \) into \( f_{A_1(G - e)}(S_1) \cdot f_{A_2(G - e, S_1)}(S_2) \) at a cost of at most \( d_{TV}(A_1(G), A_1(G - e)) \cdot H \), where \( H \) denotes the maximum Hamming weight among those of solutions obtained by running \( A \) on \( G \) and \( \{G - e\} \) over all \( e \in E \).

Thus, the earth mover’s distance between \( A(G) \) and \( A(G - e) \) is at most

\[ d_{TV}(A_1(G), A_1(G - e)) \cdot H + \int_{S_1} f_{A_1(G)}(S_1) \cdot d_{EM}(A_2(G, S_1), A_2(G - e, S_1)) \, dS_1. \]

Hence, the average sensitivity of \( A \) can be bounded as:

\[ \mathbb{E}_{e \sim E} [d_{EM}(A(G), A(G - e))] \leq H \cdot \mathbb{E}_{e \sim E} [d_{TV}(A_1(G), A_1(G - e))]

\[ + \mathbb{E}_{e \sim E} \left[ \int_{S_1 \in S_1} f_{A_1(G)}(S_1) \cdot d_{EM}(A_2(G, S_1), A_2(G - e, S_1)) \, dS_1 \right] \]

\[ \leq H \gamma_1(G) + \mathbb{E}_{S_1 \sim A_1(G)} \left[ \mathbb{E}_{e \sim E} d_{EM}(A_2(G, S_1), A_2(G - e, S_1)) \right] \]

\[ = H \gamma_1(G) + \mathbb{E}_{S_1 \sim A_1(G)} \left[ \mathbb{E}_{e \sim E} \beta_2(S_1)(G) \right]. \]

We are able to interchange the order of expectations because of Fubini’s theorem [11].
The following theorem states the composition of average sensitivity with respect to the total variation distance.

**Theorem 1.7** (Sequential composition w.r.t. the TV distance). Consider \( k \) randomized algorithms \( A_i : G \times \prod_{j=1}^{k-1} S_j \rightarrow S_i \) for \( i \in \{1, \ldots, k\} \). Suppose that, for each \( i \in \{1, \ldots, k\} \), the average sensitivity of \( A_i(\cdot, S_1, \ldots, S_{i-1}) \) is \( \gamma_i \) with respect to the total variation distance for every \( S_1 \in S_1, \ldots, S_{i-1} \in S_{i-1} \). Consider a sequence of computations \( S_1 = A_1(G), S_2 = A_2(G, S_1), \ldots, S_k = A_k(G, S_1, \ldots, S_{k-1}) \). Let \( A : G \rightarrow S_k \) be a randomized algorithm that performs this sequence of computations on input \( G \) and outputs \( S_k \). Then, the average sensitivity of \( A \) with respect to the total variation distance is at most \( \sum_{i=1}^{k} \gamma_i(G) \).

Theorem 1.7 can be immediately obtained by iteratively applying Lemma 10.1.

**Lemma 10.1.** Consider two randomized algorithms \( A_1 : G \rightarrow S_1, A_2 : G \times S_1 \rightarrow S_2 \) for a graph problem. Suppose that the average sensitivity of \( A_1 \) is \( \gamma_1(G) \) and the average sensitivity of \( A_2(\cdot, S_1) \) is \( \gamma_2(G) \) for any \( S_1 \in S_1 \), both with respect to the total variation distance. Let \( A : G \rightarrow S_2 \) be a randomized algorithm obtained by composing \( A_1 \) and \( A_2 \), that is, \( A(G) = A_2(G, A_1(G)) \). Then, the average sensitivity of \( A \) is \( \gamma_1(G) + \gamma_2(G) \) with respect to the total variation distance.

**Proof.** For a distribution \( D \), we use \( f_D \) to denote its probability mass function. Consider a graph \( G = (V, E) \). Note that

\[
f_{A_1(G)}(S_2) = \int_{S_1} f_{A_2(G,S_1)}(S_2)f_{A_1(G)}(S_1) \, dS_1.
\]

Then we have that, for \( e \in E \),

\[
d_{TV}(A(G), A(G - e))
\]

\[
= \frac{1}{2} \int_{S_2} \left| \int_{S_1} f_{A_2(G,S_1)}(S_2)f_{A_1(G)}(S_1) \, dS_1 - \int_{S_1} f_{A_2(G-e,S_1)}(S_2)f_{A_1(G-e)}(S_1) \, dS_1 \right| \, dS_2
\]

\[
= \frac{1}{2} \int_{S_2} \left| \int_{S_1} f_{A_2(G,S_1)}(S_2)\left(f_{A_1(G)}(S_1) - f_{A_1(G-e)}(S_1)\right) \, dS_1 - \int_{S_1} \left(f_{A_2(G-e,S_1)}(S_2) - f_{A_2(G,S_1)}(S_2)\right)f_{A_1(G-e)}(S_1) \, dS_1 \right| \, dS_2
\]

\[
\leq \frac{1}{2} \int_{S_1} \left| f_{A_1(G)}(S_1) - f_{A_1(G-e)}(S_1) \right| \, dS_1 \cdot \int_{S_2} f_{A_2(G,S_1)}(S_2) \, dS_2 + \int_{S_1} f_{A_1(G-e)}(S_1) \, dS_1 \cdot \frac{1}{2} \int_{S_2} \left| f_{A_2(G-e,S_1)}(S_2) - f_{A_2(G,S_1)}(S_2) \right| \, dS_2
\]

\[
= \frac{1}{2} \int_{S_1} \left| f_{A_1(G)}(S_1) - f_{A_1(G-e)}(S_1) \right| \, dS_1 + \int_{S_1} f_{A_1(G-e)}(S_1) \, dS_1 \cdot \frac{1}{2} \int_{S_2} \left| f_{A_2(G-e,S_1)}(S_2) - f_{A_2(G,S_1)}(S_2) \right| \, dS_2
\]

\[
= d_{TV}(A_1(G), A_1(G - e)) + \int_{S_1} f_{A_1(G-e)}(S_1) \cdot d_{TV}(A_2(G, S_1), A_2(G - e, S_1)) \, dS_1.
\]

Hence, the average sensitivity of \( A \) with respect to the total variation distance can be bounded as,

\[
E_{e \sim E} \left[ d_{TV}(A(G), A(G - e)) \right] \leq E_{e \sim E} \left[ d_{TV}(A_1(G), A_1(G - e)) \right] +
\]

42
\[
\mathbb{E}_{e \sim E} \left[ \int_{S_1} f_{A_1(G-e)}(S_1) \cdot d_{TV}(A_2(G, S_1), A_2(G-e, S_1)) \, dS_1 \right] \\
\leq \gamma_1(G) + \int_{S_1} f_{A_1(G-e)}(S_1) \, dS_1 \cdot \gamma_2(G) = \gamma_1(G) + \gamma_2(G).
\]

\[\square\]

10.3 Parallel composition

In this section, we prove Theorem 1.8, which bounds the average sensitivity of an algorithm obtained by running different algorithms according to a distribution in terms of the average sensitivities of the component algorithms. We restate the theorem here.

**Theorem 1.8 (Parallel composition).** Let \( A_1, A_2, \ldots, A_k \) be algorithms for a graph problem with average sensitivities \( \beta_1, \beta_2, \ldots, \beta_k \), respectively. Let \( A \) be an algorithm that, given a graph \( G \), runs \( A_i \) with probability \( \rho_i(G) \) for \( i \in [k] \), where \( \sum_{i \in [k]} \rho_i(G) = 1 \). Let \( H \) denote the maximum Hamming weight among those of solutions obtained by running \( A \) on \( G \) and \( \{G-e\}_{e \in E} \). Then the average sensitivity of \( A \) is at most \( \sum_{i \in [k]} \rho_i(G) \cdot \beta_i(G) + H \cdot \mathbb{E}_{e \sim E} \left[ \sum_{i \in [k]} |\rho_i(G) - \rho_i(G-e)| \right] \).

**Proof.** Consider a graph \( G = (V, E) \). For a solution \( S \), let \( p^G(S) \) denote the probability that \( S \) is output on input \( G \) by \( A \). Let \( p^G_e(S) \) denote the probability that \( S \) is output on input \( G \) by \( A_i \). For every solution \( S \), we know that \( p^G(S) = \sum_{i \in [k]} \rho_i(G) \cdot p^G_e(S) \).

Let \( A(G) \) denote the output distribution of \( A \) on \( G \). Fix \( e \in E \). We first bound the earth mover’s distance between \( A(G) \) and \( A(G-e) \). In order to transform \( A(G) \) into \( A(G-e) \), we first transform \( p^G(S) \), for each solution \( S \), into \( \sum_{i \in [k]} \rho_i(G) \cdot p^G_e(S) \). This can be done at a cost of at most \( \sum_{i \in [k]} \rho_i(G) \cdot d_{EM}(A_i(G), A_i(G-e)) \).

We now convert \( \sum_{i \in [k]} \rho_i(G) \cdot p^G_e(S) \), for each solution \( S \), into \( \sum_{i \in [k]} \rho_i(G-e) \cdot p^G_e(S) \) at a cost of at most \( 2H \cdot \frac{1}{2} \sum_{i \in [k]} |\rho_i(G) - \rho_i(G-e)| \), where \( \frac{1}{2} \sum_{i \in [k]} |\rho_i(G) - \rho_i(G-e)| \) is the total variation distance between the probability distributions with which \( A \) selects the algorithms on inputs \( G \) and \( G-e \). Hence, the average sensitivity of \( A \) is at most

\[
\sum_{i \in [k]} \rho_i(G) \cdot \beta_i(G) + H \cdot \mathbb{E}_{e \sim E} \left[ \sum_{i \in [k]} |\rho_i(G) - \rho_i(G-e)| \right].
\]

\[\square\]

We separately state the special case of Theorem 1.8 for \( k = 2 \).

**Theorem 10.2.** Let \( A_1 \) and \( A_2 \) be two algorithms for a graph problem with average sensitivities \( \beta_1(G) \) and \( \beta_2(G) \), respectively. Let \( A \) be an algorithm that, given a graph \( G \), runs \( A_1 \) with probability \( \rho(G) \) and runs \( A_2 \) with the remaining probability. Let \( H \) denote the maximum Hamming weight among those of solutions obtained by running \( A \) on \( G \) and \( \{G-e\}_{e \in E} \). Then the average sensitivity of \( A \) is at most \( \rho(G) \cdot \beta_1(G) + (1 - \rho(G)) \cdot \beta_2(G) + 2H \cdot \mathbb{E}_{e \sim E} [\rho(G) - \rho(G-e)] \).

10.4 Locality implies low average sensitivity

In this section, we prove Theorem 1.9 which shows that the existence of an oracle that can simulate access to the solution of a global algorithm \( A \) implies that the average sensitivity of \( A \) is bounded by the query complexity of that oracle.
Theorem 1.9 (Locality implies low average sensitivity). Consider a randomized algorithm \( A : G \to S \) for a graph problem, where each solution output by \( A \) is a subset of the set of edges of the input graph. Assume that there exists an oracle \( O \) and \( G \to S \) for a graph problem, where each solution output by \( A \) on \( G \) with \( \pi \) as its random string.

- when given access to a graph \( G = (V, E) \) and query \( e \in E \), the oracle generates a random string \( \pi \in \{0, 1\}^{r(|V|)} \) and outputs whether \( e \) is contained in the solution obtained by running \( A \) on \( G \) with \( \pi \) as its random string,
- the oracle \( O \) makes at most \( q(G) \) queries to \( G \) in expectation, where this expectation is taken over the random coins of \( A \) and a uniformly random query \( e \in E \).

Then, \( A \) has average sensitivity at most \( q(G) \). Moreover, given the promise that the input graphs satisfy \( |E| \geq |V| \), the statement applies also to algorithms for which each solution is a subset of the vertex set of the input graph.

Proof. We prove the theorem for the case that solutions output by \( A \) are subsets of edges of the input graph. It can be easily modified to work for the case that the solutions output by \( A \) are subsets of vertices of the input graph in which case, we will use the technical condition that \( n \leq m \).

Without loss of generality, assume that \( A \) uses \( r(n) \) random bits when run on graphs of \( n \) vertices. Consider a graph \( G = (V, E) \) that \( O \) gets access to. For \( e \in E \) and a string \( \pi \in \{0, 1\}^{r(n)} \), let \( Q_{e, \pi} \) denote the set of edges in \( E \) queried by \( O \) on input \( e \), while simulating the run of \( A \) with \( \pi \) as the random string. The set \( Q_{e, \pi} \) denotes the set of edges \( e' \) such that the status of \( e \) in the solutions output by \( A \) with randomness \( \pi \) on inputs \( G \) and \( G - e' \) could be different. For each edge \( e' \in E \) and string \( \pi \in \{0, 1\}^{r(n)} \), define \( R_{e', \pi} \) as the set of edges \( e \in E \) such that \( e' \in Q_{e, \pi} \).

By definition, for each \( \pi \in \{0, 1\}^{r(n)} \), we have \( \sum_{e \in E} |R_{e, \pi}| = \sum_{e \in E} |Q_{e, \pi}| \). Hence we have:

\[
\sum_{\pi \in \{0, 1\}^{r(n)}} \sum_{e \in E} |R_{e, \pi}| = \sum_{\pi \in \{0, 1\}^{r(n)}} \sum_{e \in E} |Q_{e, \pi}|,
\]

and

\[
\mathbb{E}_{\pi \in \{0, 1\}^{r(n)}} \mathbb{E}_{e \sim E} |R_{e, \pi}| \leq \mathbb{E}_{\pi \in \{0, 1\}^{r(n)}} \mathbb{E}_{e \sim E} |Q_{e, \pi}| \leq q(G),
\]

where the last inequality follows from our assumption on \( O \).

For \( \pi \in \{0, 1\}^{r(n)} \) and \( e \in E \), the set \( R_{e, \pi} \) contains the set of edges whose presence in the solution could be affected by the removal of \( e \) from \( G \). Therefore, it is a superset of the set of edges contained in the symmetric difference between the outputs of \( A \) on inputs \( G \) and \( G - e \) when run with \( \pi \) as the random string.

Let \( \mathcal{H}_{A, \pi}(G, G') \) denote the Hamming distance between the outputs of the algorithm \( A \) on inputs \( G \) and \( G' \) when run with \( \pi \) as the random string. As per this notation, for each \( e \in E \),

\[
\mathbb{E}_{\pi \in \{0, 1\}^{r(n)}} \mathcal{H}_{A, \pi}(G, G - e) \leq \mathbb{E}_{\pi \in \{0, 1\}^{r(n)}} |R_{e, \pi}|.
\]

The following claim relates the quantity on the left hand side of the above inequality with the average sensitivity of \( A \).

\footnote{If \( r(G) \) is the length of the random string used for \( G \), we can simply set \( r(n) = \max\{r(G) : G = (V, E), |V| = n\} \). If we do not need \( r(n) \) bits for some particular graph \( G \) on \( n \) vertices, we can just throw away the unused bits.}
Claim 10.3. The average sensitivity of $A$ is bounded as
\[
\beta(G) \leq \mathbb{E}_{e \in E(G)} \mathbb{E}_{\pi \in \{0,1\}^{|V|}} \mathcal{H}_{A,\pi}(G, G-e).
\]

Proof. Fix $G \in \mathcal{G}$ and $e \in E(G)$. We first bound the earth mover's distance between $A(G)$ and $A(G-e)$, where $A(G)$ and $A(G-e)$ are the output distributions of $A$ on inputs $G$ and $G-e$, respectively. For $S \in \mathcal{S}$, let $p_G(S)$ and $p_{G-e}(S)$ denote the probabilities that $A$ outputs $S$ on $G$ and $G-e$, respectively. We start with $A(G)$. Consider a string $\pi \in \{0,1\}^{|V|}$. Let $S \in \mathcal{S}$ denote the output of $A$ on input $G$ when using the string $\pi$ as its random string. Let $S'$ denote the output that is generated when running $A$ on input $G-e$ with $\pi$ as the random string. We move a mass of $\frac{1}{2^{r(n)}}$ (corresponding to the string $\pi$) from $p_G(S)$ to $p_G(S')$ at a cost of $\frac{d_{Ham}(S,S')}{2^{|V|}}$. Moving masses corresponding to every string $\pi \in \{0,1\}^{|V|}$ this way, we can transform $A(G)$ to $A(G-e)$. The total cost incurred during this transformation is $\mathbb{E}_{\pi \in \{0,1\}^{|V|}} \mathcal{H}_{A,\pi}(G, G-e)$. Therefore the earth mover's distance between $A(G)$ and $A(G-e)$ is at most $\mathbb{E}_{\pi \in \{0,1\}^{|V|}} \mathcal{H}_{A,\pi}(G, G-e)$. Therefore the average sensitivity of $A$ is $\beta(G) \leq \mathbb{E}_{e \in E(G)} \mathbb{E}_{\pi \in \{0,1\}^{|V|}} |R_{e,\pi}| \leq q(G)$. \hfill \qed

Acknowledgments. We are grateful to anonymous reviewers for suggesting a major improvement to the average sensitivity analysis of Algorithm [6]. We thank Tasuku Soma for several helpful discussions. We extend our gratitude to Sofya Raskhodnikova for helpful comments that improved the presentation of this article.

References


A Average Sensitivity of Exponential Mechanism

In this section, we prove Lemma 2.1.
Proof of Lemma 2.4. Let \( t > 0 \) be a parameter. Note that any index \( i \in [n] \) with \( x(i) > \text{OPT} + \log n/\eta + t/\eta \) has probability at most \( e^{-t}/n \) of being sampled by \( A \). Hence, by a union bound, for every \( t > 0 \)

\[
\Pr_{i \sim A(x)} \left[ x(i) \geq \text{OPT} + \log n/\eta + t/\eta \right] \leq e^{-t}.
\]

Next, we analyze the distance between the output distributions. Let \( x, x' \in \mathbb{R}^n \) be vectors, and let \( Z = \sum_{i \in [n]} e^{-\eta x(i)} \) and \( Z' = \sum_{i \in [n]} e^{-\eta x'(i)} \). Without loss of generality, we assume that \( Z \geq Z' \). First, note that for all \( i \in [n] \) such that \( x(i) \geq x'(i) \), we have

\[
0 \leq e^{-\eta x'(i)} - e^{-\eta x(i)} = e^{-\eta x'(i)} \left( 1 - e^{-\eta (x(i) - x'(i))} \right) \leq \eta e^{-\eta x'(i)} (x(i) - x'(i)).
\]

Hence for any \( i \in [n] \), we have

\[
|e^{-\eta x(i)} - e^{-\eta x'(i)}| \leq \max \{ \eta e^{-\eta x'(i)} (x'(i) - x(i)), \eta e^{-\eta x'(i)} (x(i) - x'(i)) \}
\leq \eta |x(i) - x'(i)| \max \{ e^{-\eta x(i)}, e^{-\eta x'(i)} \} \leq \eta |x(i) - x'(i)| \left( e^{-\eta x(i)} + e^{-\eta x'(i)} \right).
\]

Then, we have

\[
\frac{1}{Z} \sum_{i \in [n]} |e^{-\eta x(i)} - e^{-\eta x'(i)}| \leq \frac{\eta}{Z} \sum_{i \in [n]} |x(i) - x'(i)| \left( e^{-\eta x(i)} + e^{-\eta x'(i)} \right)
\leq \frac{\eta}{Z} \max_{i \in [n]} |x(i) - x'(i)| \sum_{i \in [n]} \left( e^{-\eta x(i)} + e^{-\eta x'(i)} \right) = \frac{\eta(Z + Z')}{Z} \max_{i \in [n]} |x(i) - x'(i)|
\leq 2\eta \|x - x'\|_1.
\]

Then, the total variation distance between \( A(x) \) and \( A(x') \) is at most

\[
\sum_{i \in [n]} \left| \frac{\exp(-\eta x(i))}{Z} - \frac{\exp(-\eta x'(i))}{Z'} \right| = \sum_{i \in [n]} \left| \frac{\exp(-\eta x(i))}{Z} - \frac{\exp(-\eta x'(i))}{Z} \right| \left( \frac{Z - Z' + Z'}{Z'} \right)
\leq \frac{1}{Z} \sum_{i \in [n]} \left| e^{-\eta x(i)} - e^{-\eta x'(i)} \right| + \frac{Z - Z'}{Z} \frac{1}{Z} \sum_{i \in [n]} \exp(-\eta x'(i))
\leq \frac{2}{Z} \sum_{i \in [n]} \left| e^{-\eta x(i)} - e^{-\eta x'(i)} \right| \leq 4\eta \|x - x'\|_1.
\]

B Average Sensitivity of Prim’s algorithm

In this section, we show that Prim’s algorithm (with a simple tie-breaking rule, as described in Algorithm 9) has high average sensitivity even on unweighted graphs. This is in contrast to the low average sensitivity of Kruskal’s algorithm that we discussed in Section 3.

Lemma B.1. The average sensitivity of Prim’s algorithm is \( \Omega(m) \).
Algorithm 9: Prim’s Algorithm

Input: undirected graph \( G = ([n], E) \)

1. Let \( T \leftarrow \{1\} \);
2. while there exists a vertex not spanned by \( T \) do
3.   Let \( E' \) be the set of edges with the smallest weight among all the edges in \( E \) that have exactly one endpoint in \( T \);
4.   Add to \( T \), an edge from \( E' \) that has lexicographically smallest \( T \)-endpoint among all edges in \( E' \), breaking further ties arbitrarily.

5. return Output \( T \).

---

**Figure 1:** The graph family \( \{G_n\}_{n \in 2\mathbb{N}} \).

**Proof.** Consider the graph family \( \{G_n\}_{n \in 2\mathbb{N}} \) in Figure 1. For a large enough \( n \in 2\mathbb{N} \), consider running Algorithm 9 on \( G_n \). The tree \( T \) output will consist of the edges \((i, i+1)\) for all \( i \in [n/2-2]\), the edges \((n/2-1, j)\) for all \( j \in \{n/2+1, \ldots, n\} \), and the edge \((n/2, 1)\).

If we remove an edge \((i', i'+1)\) for \( i' \in [n/2-2] \) from \( G_n \) and run Algorithm 9 on the resulting graph, the tree, say \( T_{i'} \), output will consist of all edges of the form \((i, i+1)\) for \( i \in [n/2-1] \setminus \{i'\} \), all edges of the form \((n/2, j)\) for all \( j \in \{n/2+1, \ldots, n\} \), and the edges \((n/2+1, n/2-1)\) and \((n/2, 1)\). The Hamming distance of \( T_{i'} \) from \( T \) is equal to \( n/2 \).

Since a uniformly random edge removed from \( G_n \) is of the form \((i, i+1)\) for \( i \in [n/2-2] \) with probability \( \frac{n/2-2}{3n/2-1} \), the average sensitivity of Algorithm 9 is at least \( \frac{n}{2} \cdot \frac{n/2-2}{3n/2-1} \), which is at least \( \frac{n}{6} - 1 = \Omega(m) \) for the family \( \{G_n\}_{n \in 2\mathbb{N}} \). \( \square \)

**C Linear Programming: Algorithm and Analysis**

In this section, we prove Theorem 5.1. First in Section C.1 we consider an LP solver based on sampling from a Gibbs distribution, whose average sensitivity depends on the optimal value. Then in Section C.2 we get rid of the dependency and provide an LP solver with average sensitivity \( O(\log n/\epsilon) \).
The running time is polynomial in $n$ where we used the fact that

Moreover, the average sensitivity of $A$ is

The running time is polynomial in $n$ and $e^\eta$.

The algorithm $A$ in Theorem C.1 is very simple: Given a cost vector $c \in [0,1]^n$, we draw a sample from a distribution $D$ with $d_{TV}(D, D_{\eta,c,K}) \leq \epsilon$ in time polynomial in $n$, $e^\eta$, and $\log(1/\epsilon)$. Then, there exists an algorithm $A$ that, given a cost vector $c \in [0,1]^n$ such that $\text{OPT} = \text{OPT}(c) \geq 1$, outputs $x \in K$ with

Moreover, the average sensitivity of $A$ is

The running time is polynomial in $n$ and $e^\eta$.

In this section, we fix $0 < \eta \leq 1$, $c \in [0,1]^n$, and $K \subseteq [0,1]^n$.

Throughout this section, we consider the following.

\begin{algorithm}
\textbf{Input:} $0 < \eta \leq 1$, a vector $c \in [0,1]^n$, and a convex polytope $K \subseteq [0,1]^n$.
\begin{enumerate}
\item Sample $x \in K$ from a distribution $D$ with $d_{TV}(D, D_{\eta,c,K}) \leq 1/n$;
\item return $x$.
\end{enumerate}
\end{algorithm}

C.1 LP solver based on sampling from Gibbs distributions

In this section, we analyze the performance of Algorithm 10 and prove Theorem C.1. Throughout this section, we fix $0 < \eta \leq 1$, $c \in [0,1]^n$, and $K \subseteq [0,1]^n$.

Let $D_0 := D_{\eta,c,K}$ and $Z_0 := Z_{\eta,c,K}$. For $i \in [n]$, let $D_i := D_{\eta,c^{-i},K}$. Specifically, its probability density function is $f_{\eta,c^{-i},K}(x) := \exp(-\eta(c^{-i},x))/Z_i$, where $Z_i = \int_K \exp(-\eta(c^{-i},x))dx$. Note that $Z_i \geq Z_0$ for all $i \in [n]$ holds since $\langle c^{-i}, x \rangle \leq \langle c, x \rangle$ for all $i \in [n], x \in K$. First, we bound the expected value of the inner product $\langle c, x \rangle$ when $x$ is sampled from $D_0$.

\begin{lemma}
We have

$$\mathbb{E}_{x \sim D_{\eta,c,K}} \langle c, x \rangle \leq \text{OPT} + \frac{\log n}{\eta} + 1.$$

\end{lemma}

\begin{proof}
Let $A = \{x \in K \mid \langle c, x \rangle \geq \text{OPT} + t\}$, where $t$ is a parameter determined later. Then, we have

$$\mathbb{E}_{x \sim D_0} \langle c, x \rangle \leq \int_A e^{-\eta(\text{OPT}+t)}dx \leq e^{-\eta\text{OPT}/e^\eta t} \leq 1/e^\eta t,$$

where we used the fact that $\int_A dx \leq 1$ as $A \subseteq K \subseteq [0,1]^n$. Hence, we have

$$\mathbb{E}_{x \sim D_0} \langle c, x \rangle \leq \mathbb{P}_{x \sim D_0} [x \not\in A] \cdot (\text{OPT} + t) + \mathbb{P}_{x \sim D_0} [x \in A] \cdot n \leq \text{OPT} + t + n/e^\eta t.$$
By taking $t = \log(n)/\eta$, we have

$$\mathbb{E}_{x \sim \mathcal{D}_0} \langle c, x \rangle \leq \text{OPT} + \frac{\log(n)}{\eta} + 1.$$ \hfill \Box

Now, we analyze the average sensitivity of Algorithm 10. We start with the following lemma, which states that $Z_0$ and $Z_i$ are close on average over $i \in [n]$.

**Lemma C.3.** We have

$$\frac{1}{Z_0} \sum_{i \in [n]} (Z_i - Z_0) = O(\eta \text{OPT} + \log n).$$

**Proof.** We have

$$\frac{1}{Z_0} \sum_{i \in [n]} (Z_i - Z_0) = \frac{1}{Z_0} \sum_{i \in [n]} \int_K \left( e^{-\eta(c^{-1}, x)} - e^{-\eta(c, x)} \right) dx = \frac{1}{Z_0} \int_K \sum_{i \in [n]} e^{\eta(c(i)x(i))} - 1 - e^{-\eta(c, x)} dx$$

$$\leq \frac{(e - 1)\eta}{Z_0} \int_K \sum_{i \in [n]} c(i)x(i) - e^{-\eta(c, x)} dx \quad \text{(by } e^x \leq 1 + (e - 1)x \text{ for } x \in [0, 1])$$

$$= \frac{(e - 1)\eta}{Z_0} \int_K \langle c, x \rangle - e^{-\eta(c, x)} dx$$

$$\leq (e - 1)\eta \left( \text{OPT} + \frac{\log(n)}{\eta} + 1 \right), \quad \text{(by Lemma C.2)}$$

as desired. \hfill \Box

Next, we bound the average earth mover’s distance between $\mathcal{D}_0$ and $\mathcal{D}_i$, where the average is over $i \in [n]$ and the underlying distance metric is the $\ell_1$ distance.

**Lemma C.4.** We have

$$\mathbb{E}_{i \sim [n]} d_{\text{EM}}^1(\mathcal{D}_0, \mathcal{D}_i) = O(\eta \text{OPT} + \log n).$$

**Proof.** We have

$$\mathbb{E}_{i \sim [n]} d_{\text{EM}}^1(\mathcal{D}_0, \mathcal{D}_i) \leq \frac{1}{n} \sum_{i \in [n]} \int_K \left( \|x\|_1 \cdot \left| \frac{\exp(-\eta(c^{-1}, x))}{Z_i} - \frac{\exp(-\eta(c, x))}{Z_0} \right| \right) dx$$

$$\leq \sum_{i \in [n]} \int_K \left| \frac{\exp(-\eta(c^{-1}, x))}{Z_i} \right| \left( 1 - \frac{Z_i - Z_0}{Z_i} \right) - \frac{\exp(-\eta(c, x))}{Z_0} dx$$

$$\leq \sum_{i \in [n]} \int_K \frac{\exp(-\eta(c^{-1}, x))}{Z_0} - \frac{\exp(-\eta(c, x))}{Z_0} dx + \sum_{i \in [n]} \int_K \frac{\exp(-\eta(c^{-1}, x))}{Z_0} \frac{Z_i - Z_0}{Z_i} dx$$

$$\leq \frac{2}{Z_0} \sum_{i \in [n]} (Z_i - Z_0)$$

$$= O(\eta \text{OPT} + \log n), \quad \text{(by Lemma C.3)}$$

as desired. \hfill \Box
Proof of Theorem C.1 Let $\mathcal{A}$ be Algorithm 10. By running the algorithm given in the premise of Theorem C.1, we can sample in polynomial time from a distribution $\mathcal{D}$ such that $d_{TV}(\mathcal{D}, \mathcal{D}_0) \leq 1/n$. Then by Lemma C.2 we have

$$\mathbb{E}_{x \sim \mathcal{A}} \langle c, x \rangle \leq \mathbb{E}_{x \sim \mathcal{D}_0} \langle c, x \rangle + d_{TV}(\mathcal{D}, \mathcal{D}_0) \cdot n \leq \text{OPT} + \frac{\log n}{\eta} + 1.$$

By Lemma C.4 the average sensitivity of $\mathcal{A}$ is at most

$$\mathbb{E}_{i \sim [n]} d_{\text{EM}}(\mathcal{A}(c), \mathcal{A}(c^{-i})) \leq \mathbb{E}_{i \sim [n]} [\varphi_{\mathcal{D}_0}(\mathcal{D}_0) + d_{TV}(\mathcal{A}(c), \mathcal{D}_0) \cdot n + d_{TV}(\mathcal{A}(c^{-i}), \mathcal{D}_i) \cdot n] = O(\eta \text{OPT} + \log n + 2)$$

as desired. □

C.2 LP solver with logarithmic average sensitivity

In this section, we prove Theorem 5.1. One may think that by setting $\eta = \min\{\log n/\text{OPT}, 1\}$ in Theorem C.1, we can get an algorithm with the expected cost $(1 + \epsilon)\text{OPT} + O(\log n)$ and average sensitivity $O(\log n)$. However, for Theorem C.1 to hold, the parameter $\eta$ should be a constant independent of $c$. To get around this issue, we consider an algorithm that samples $0 \leq \eta \leq 1$ from a certain distribution and then apply the algorithm in Theorem C.1.

Let $\text{Lap}(\mu, \phi)$ denote the Laplace distribution with a location parameter $\mu$ and a scale parameter $\phi$. Let $f_{\mu, \phi}(x) := \frac{1}{2\phi} \exp(-|x - \mu|/\phi)$ be the probability density function of $\text{Lap}(\mu, \phi)$. Then, our algorithm first samples $\eta$ from the distribution $\text{Lap}(\tau, \delta \tau/2)$, where $\tau := \min\{\log n / \epsilon \text{OPT}, 2/3\}$ and $\delta = 1/\log n$. If $0 < \eta \leq 1$ holds, then it runs Algorithm 10 with $\eta$ and returns the output. Otherwise, it simply computes an arbitrary optimal solution to the LP $\min_{x \in K} \langle c, x \rangle$, and then returns it. The pseudocode is given in Algorithm 11. The time complexity is clearly polynomial in $n$ because we run Algorithm 10 only when $\eta$ is at most one. In the next section, we prove Theorem 5.1 by analyzing the solution quality and the average sensitivity of Algorithm 11.

C.2.1 Analysis

In this section, we analyze the performance of Algorithm 11. Throughout this section, we fix $\epsilon > 0$, $c \in [0, 1]^n$, and $K \subseteq [0, 1]^n$. We use the symbol $\mathcal{A}$ to denote Algorithm 11.

First, we analyze the solution quality of Algorithm 11. Recalling that the cumulative density function of $\text{Lap}(\mu, \phi)$ is $\exp((x - \mu)/\phi)/2$ for $x < \mu$ and $1 - \exp(-(x - \mu)/\phi)/2$ for $x \geq \mu$, the following is immediate.
Proposition C.5. The probability that $L \sim \text{Lap}(\tau, \delta \tau/2)$ falls in $[-\infty, \tau/2] \cup [3\tau/2, \infty]$ is at most $\exp(-1/\delta)$.

Lemma C.6. \[ \mathbb{E}_{x \sim A(\epsilon, c, K)} [(c, x)] \leq (1 + 2\epsilon)\text{OPT} + O(\log n). \]

Proof. By Proposition C.5 \[ \tau/2 \leq \eta \leq 3\tau/2 \leq 1 \] holds with probability at least $1 - \exp(-1/\delta)$. Then by Theorem C.1, we have

\[
\begin{align*}
\mathbb{E}_{x \sim A(\epsilon, c, K)} [(c, x)] & \leq \int_{\tau/2}^{3\tau/2} \left( \text{OPT} + \log \frac{n}{\eta} + 1 \right) f_{\tau, \delta \tau/2} (\eta) d\eta + e^{-1/\delta} \cdot n \\
& \leq \int_{\tau/2}^{3\tau/2} \left( \text{OPT} + \frac{2\log n}{\tau} + 1 \right) f_{\tau, \delta \tau/2} (\eta) d\eta + e^{-1/\delta} \cdot n \\
& \leq \text{OPT} + \frac{2\log n}{\tau} + 2 \leq \text{OPT} + \max \{2\epsilon \text{OPT}, 3 \log n\} + 2 \\
& \leq (1 + 2\epsilon)\text{OPT} + 3 \log n + 2.
\end{align*}
\]

Next, we analyze the average sensitivity of Algorithm 11. For $i \in [n]$, let $\text{OPT}_i := \text{OPT}(e^{-i})$ and $\tau_i := \min \{ \log n / \epsilon \text{OPT}_i, 2/3 \}$. Let $\Delta_\tau = \max_{i \in [n]} \tau_i - \tau$. Note that $\Delta_\tau \geq 0$ holds. The following fact is a restatement of Claim 8.9.

Lemma C.7. Let $\eta \sim \text{Lap}(\tau, \delta \tau), i \in [n]$, and $\eta_i \sim \text{Lap}(\tau_i, \delta \tau_i)$. Then, we have

\[
d_{TV}(\eta, \eta_i) \leq O \left( \frac{\Delta_\tau}{\delta (\tau - \Delta_\tau)} + e^{-1/\delta} \right).
\]

We can bound $\Delta_\tau / (\tau - \Delta_\tau)$ in terms of $\Delta_{\text{OPT}}$ as follows.

Lemma C.8. We have

\[
\frac{\Delta_\tau}{\tau - \Delta_\tau} \leq \frac{1}{\text{OPT} - 2}.
\]

Proof. Let $i^* \in [n]$ denote $\arg \max_{i \in [n]} \tau_i$. Then,

\[
\begin{align*}
\Delta_\tau &= \max_{i \in [n]} \tau_i - \tau \leq \min \left\{ \log n / \epsilon \text{OPT}_i^*, 2 \right\} - \min \left\{ \log n / \epsilon \text{OPT}^* \right\} \\
& \leq \log n / \epsilon \text{OPT}_i^* - \log n / \epsilon \text{OPT}^* \quad \text{(by } \text{OPT}_i^* \leq \text{OPT}) \\
& \leq \log n \cdot \frac{\text{OPT} - \text{OPT}_i^*}{\epsilon \text{OPT} \cdot \text{OPT}_i^*} \\
& \leq \log n / \epsilon \cdot \frac{1}{\text{OPT}(\text{OPT} - 1)}. \quad \text{(by } \text{OPT} \leq \text{OPT}_i^* + 1)
\end{align*}
\]

Then, we have

\[
\frac{\Delta_\tau}{\tau - \Delta_\tau} = \frac{\Delta_\tau}{\tau} \cdot \frac{\tau}{\tau - \Delta_\tau} \leq \frac{1}{\text{OPT} - 1} \cdot \frac{1}{1 - \frac{1}{\text{OPT} - 1}} \leq \frac{1}{\text{OPT} - 1} \cdot \frac{\text{OPT} - 1}{\text{OPT} - 2} = \frac{1}{\text{OPT} - 2}.
\]
Lemma C.9. The average sensitivity of Algorithm II is
\[ O\left(\frac{\log n}{\epsilon}\right). \]

Proof. Let \( A \) be Algorithm II. We first bound the earth mover’s distance \( d_E^A(A(c), A(c^{-i})) \) for \( i \in [n] \). For \( \eta \in \mathbb{R} \), let \( A_\eta \) be the algorithm that runs Algorithm I with \( \eta \) if \( 0 < \eta \leq 1 \) and runs an arbitrary LP solver otherwise. Then, we consider an algorithm \( A(c^{-i}) \) that samples \( \eta \) from the distribution \( \text{Lap}(\tau, \delta\tau) \), instead of \( \text{Lap}(\tau, \delta\tau) \), and then runs \( A_\eta(c^{-i}) \). Then, we have for all \( i \in [n] \)
\[ d_E^A(A(c), A(c^{-i})) \leq d_E^A(A(c), A'(c^{-i})) + d_E^A(A(c^{-i}), A'(c^{-i})). \]

For the first term, we have
\[
\begin{align*}
&d_E^A(A(c), A'(c^{-i})) \leq \int_{-\infty}^{\infty} d_E^A(A_\eta(c), A_\eta(c^{-i})) f_{\tau, \delta\tau}(\eta) d\eta \\
&\leq e^{-1/\delta} \cdot n + \int_{\tau/2}^{3\tau/2} d_E^A(A_\eta(c), A_\eta(c^{-i})) f_{\tau, \delta\tau}(\eta) d\eta \quad \text{(by Proposition C.5)} \\
&\leq 1 + \int_{\tau/2}^{3\tau/2} d_E^A(A_\eta(c), A_\eta(c^{-i})) f_{\tau, \delta\tau}(\eta) d\eta.
\end{align*}
\]

For the second term, we have
\[
\begin{align*}
&d_E^A(A(c^{-i}), A'(c^{-i})) \leq \int_{-\infty}^{\infty} \left( \mathbb{E}_{x \sim A_\eta(c^{-i})} \|x\|_1 \cdot |f_{\tau, \delta\tau}(\eta) - f_{\tau, \delta\tau}(\eta)| \right) d\eta \\
&\leq e^{-1/\delta} \cdot n + \int_{\tau/2}^{3\tau/2} \left( \mathbb{E}_{x \sim A_\eta(c^{-i})} \|x\|_1 \cdot |f_{\tau, \delta\tau}(\eta) - f_{\tau, \delta\tau}(\eta)| \right) d\eta \quad \text{(by Proposition C.5)} \\
&\leq e^{-1/\delta} \cdot n + \int_{\tau/2}^{3\tau/2} \left( \text{OPT} + O\left(\frac{\log n}{\eta}\right) \right) |f_{\tau, \delta\tau}(\eta) - f_{\tau, \delta\tau}(\eta)| d\eta \quad \text{(by Theorem C.1)} \\
&\leq e^{-1/\delta} \cdot n + \left( \text{OPT} + O\left(\frac{\log n}{\tau}\right) \right) \int_{\tau/2}^{3\tau/2} |f_{\tau, \delta\tau}(\eta) - f_{\tau, \delta\tau}(\eta)| d\eta \quad \text{(by Lemma C.7)} \\
&\leq 1 + \left( (1 + O(\epsilon))\text{OPT} + O(\log n) \right) \cdot \left( O\left(\frac{\Delta_\tau}{\delta_2 (\tau - \Delta_\tau)}\right) \right) \quad \text{(by Lemma C.8)} \\
&= O\left( \frac{\text{OPT} + \log n}{\text{OPT} - 2} \right).
\end{align*}
\]

Hence, we have
\[
d_E^A(A(c), A(c^{-i})) = O\left( \int_{\tau/2}^{3\tau/2} d_E^A(A_\eta(c), A_\eta(c^{-i})) f_{\tau, \delta\tau}(\eta) d\eta + \frac{\text{OPT} + \log n}{\text{OPT} - 2} \right).
\]

The average sensitivity of Algorithm III can be bounded as
\[
\mathbb{E}_{i \sim [n]} d_E^A(A(c), A(c^{-i})).
\]
\[
O \left( \int_{\tau/2}^{3\tau/2} \mathbb{E}_{i \sim [n]} d_{\text{EM}}(A_\eta(c), A_\eta(c^{-i})) f_{\tau,\delta}(\eta) d\eta + \frac{\text{OPT} + \log n}{\text{OPT} - 2} \right)
\]

\[
= O \left( \int_{\tau/2}^{3\tau/2} \eta \text{OPT} + \log n f_{\tau,\delta}(\eta) d\eta + \frac{\text{OPT} + \log n}{\text{OPT} - 2} \right) \quad \text{(by Theorem C.1)}
\]

\[
= O \left( \tau \text{OPT} + \log n + \frac{\text{OPT} + \log n}{\text{OPT} - 2} \right) = O \left( \frac{\log n}{\epsilon} \right).
\]

Theorem 5.1 follows by combining Lemma C.6 and Lemma C.9 and replacing \( \epsilon \) with \( \epsilon/C \) for a sufficiently large constant \( C > 0 \).

## D Sampling from Gibbs Distributions over Metric Polytope

We define a polytope \( P \subseteq [0,1]^{(V)} \) to be

\[
P = \left\{ \mathbf{d} \in [0,1]^{(V)} \mid \mathbf{d}(u,v) + \mathbf{d}(v,w) \geq \mathbf{d}(u,w) \forall \{u,w\} \in \binom{V}{3} \right\}.
\]

as the polytope consisting of vectors \( \mathbf{d} \in [0,1]^{(V)} \) satisfying triangle inequalities. For a vector \( \mathbf{c} \in [0,1]^{(V)} \), let \( D_{\eta,c,P} \) be the distribution over \( P \), where the probability density at \( \mathbf{x} \in P \) is proportional to \( \exp(-\eta \langle \mathbf{c}, \mathbf{x} \rangle) \). The goal of this section is to show the following.

**Theorem D.1.** For \( \epsilon > 0 \) and \( \mathbf{c} \in [0,1]^{(V)} \), we can draw a sample from a distribution \( \tilde{D}_{\eta,c,P} \) with \( d_{\text{TV}}(\tilde{D}_{\eta,c,P}, D_{\eta,c,P}) \leq \epsilon \) in time polynomial in \( n := |V|, e^n, \) and \( \log(1/\epsilon) \).

At the end of this section, we briefly argue that we can get similar results for the polytopes given by the constraints in LPs (8) and (9).

For a set \( K \subseteq \mathbb{R}^n \), let \( D_K \) be the uniform distribution over \( K \). The following theorem states that, under some conditions, we can efficiently sample points from \( D_K \).

**Theorem D.2** (Corollary 1.2 of [22], rephrased). Let \( K \) be a convex body that contains a ball of radius \( r \) and is contained in a ball of radius \( R \). Suppose that a point in \( K \) at distance at least \( d \) from the boundary of \( K \) is explicitly given. Then for \( \epsilon > 0 \), we can draw a sample from a distribution \( \tilde{D}_K \) with \( d_{\text{TV}}(\tilde{D}_K, D_K) \leq \epsilon \) in \( O \left( \frac{n^3 R^2 \log R}{r^2} \right) \) time.

**Lemma D.3.** For \( \epsilon > 0 \), we can draw a sample from a distribution \( D_P \) with \( d_{\text{TV}}(D_P, D_P) \leq \epsilon \) in \( O \left( n^5 \log \frac{n}{\epsilon} \right) \) time, where \( n := |V| \).

**Proof.** We simply apply Theorem D.2 to the polytope \( P \). We have \( r \geq 1/4 \) as \( [1/2,1]^{(V)} \subseteq P \), and \( R \leq \sqrt{\binom{n}{2}}/2 \) as \( P \subseteq [0,1]^{(V)} \). Also, the point \( (3/4,\ldots,3/4) \in P \) has distance \( 1/4 \) from the boundary of \( P \). Hence, the time complexity for sampling a point from \( D_P \) is

\[
O \left( \frac{n^3 R^2 \log R}{r^2} \right) = O \left( n^5 \log \frac{n}{\epsilon} \right). \quad \square
\]
Let $D$ be a distribution over $\mathbb{R}^n$. Abusing notation, we also use $D$ to denote its probability density function. For a set $S \subseteq \mathbb{R}^n$, we define $D(S) = \int_S D(x) \, dx$ as the probability that a point in $S$ is sampled from $D$. A level set of $D$ is a set of the form $\{x \in \mathbb{R}^n \mid D(x) \geq \tau\}$ for some $\tau \in \mathbb{R}$. We use the following theorem to prove Theorem [D.1]

**Theorem D.4** (Theorem 1.3 of [22], rephrased). Let $K \subseteq \mathbb{R}^n$ be a convex body contained in a ball of radius $R$ and let $D$ be a probability distribution over $K$, where $D(x)$ for a point $x \in K$ is proportional to $e^{-\langle a, x \rangle}$ for some vector $a \in \mathbb{R}^n$. Assume that the level set of $D$ of probability $1/8$ contains a ball of radius $r$. For $\epsilon > 0$, suppose that we have sample access to a distribution $D_0$ such that $D_0(x)/D(x) \leq M$ except on a set $S \subseteq \mathbb{R}^n$ with $D_0(S) \leq \epsilon/2$. Then we can draw a sample from a distribution $\bar{D}$ with $\text{d}_{TV}(\bar{D},D) \leq \epsilon$ by drawing $\bar{D}(x) = \frac{1}{\text{vol}(P)} \cdot \int_P e^{-\eta(c, y)} \, dy - e^{\eta(c, x)}.

Hence, we have $M \leq e^{\eta(c, x)} \leq e^m$.

Next, we bound $r$ from below. Let $L$ be the level set of density $1/8$ in $P$. Then, we have

$$\text{vol}(L) \geq \frac{\text{vol}(P) \cdot e^{-\eta(n)}}{8}$$

because $e^{-\eta(c, x)}$ takes a value in $[e^{-\eta(n)}, 1]$. Let $\gamma \in [0, 1]$ be the maximum value such that the point $(\gamma, \ldots, \gamma)$ belongs to $L$. As $c \geq 0$, the polytope $\min\{2\gamma, 1\}^{(V)}$ contains $L$, and hence we have

$$\text{vol}(P) \cdot \min\{2\gamma, 1\}^{(V)} \geq \text{vol}(L) \geq \frac{\text{vol}(P) \cdot e^{-\eta(n)}}{8},$$

and it follows that $\gamma \geq \min\{e^{-\eta}/16, 1/2\} = e^{-\eta}/16$. As the polytope $L$ contains $\gamma P$, in particular, $L$ contains a hypercube $[\gamma/2, \gamma]^{(V)}$, which implies $r \geq \gamma/4 \geq e^{-\eta}/64\epsilon$. Hence, the number of samples drawn from $D_0$ is

$$O\left(\frac{n^2 R^2}{r^2} \cdot \ln^5 \frac{MnR}{r\epsilon}\right) = O\left(n^3 e^{2n} \ln^5 \frac{n^{3/2} e^{\eta(n+1)}}{\epsilon}\right) = O\left(n^3 e^{2n} \left(\eta^5 (n+1)^5 + \ln^5 \frac{n^{3/2}}{\epsilon}\right)\right),$$

which is polynomial in $n$, $e^\eta$, and $\log(1/\epsilon)$. Recalling that each draw from $D_0$ takes $O(n^5 \log(n/\epsilon))$ time, the time complexity to draw a sample from $D_{\eta, c, P}$ is polynomial in $n$, $e^\eta$, and $\log(1/\epsilon)$. 

Now, we briefly argue that we can also efficiently draw a sample from $D_{\eta, c, Q}$, where $Q$ is the polytope given by the constraints in LP (8) or (9). Indeed, for LP (8), the same analysis goes through. The only difference is that the domain is now $\binom{V}{2} - \{s, t\}$ instead of $\binom{V}{2}$. For LP (9),
we can show that $\gamma \geq e^{-\eta}/16$ as in the proof of Theorem D.1. However, as LP (9) has a lower bound on $\sum_{(u,v) \in (V_2)} d(u,v)$, the polytope $L$ may not contain the hypercube $[\gamma/2, \gamma]^{\binom{n}{2}}$; we can only guarantee that it contains the hypercube $[\alpha(1 - \alpha) n^2 / \binom{n}{2}, \gamma]^{\binom{n}{2}}$. Hence, assuming $n$ is large enough, $L$ contains $[4\alpha(1 - \alpha), \gamma]^{\binom{n}{2}}$. Now, we have

$$r \geq \frac{\gamma - 4\alpha(1 - \alpha)}{2} \geq \frac{e^{-\eta} - 64\alpha(1 - \alpha)}{32}.$$  

As $\eta \geq \log n/\epsilon \text{OPT}$ with high probability by Algorithm 11 and Proposition C.5, we have

$$r \geq \frac{n^{-1/\epsilon \text{OPT}} - 64\alpha(1 - \alpha)}{32},$$

which is constant as long as $\text{OPT} = \Omega(\log n)$.  

57