Average Sensitivity of Graph Algorithms

Nithin Varma
Boston University
nvarma@bu.edu

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

Abstract

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. Hence, it is desirable to use algorithms that, even when only a (large) subgraph is available, output solutions that are close to the solutions output when the whole graph is available. We formalize this idea 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 composability, 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 maximum matching problem, and the minimum vertex cover problem. We also show that every algorithm for the 2-coloring problem has average sensitivity linear in the number of vertices. To show our algorithmic results, we 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 [New04, New06], ranking nodes [PBMW99], and spreading influence [KKT03], the solutions output by an algorithm on G' should be close to those output on G.

We assume that the input graph G' at hand is a randomly chosen (large) subgraph of an unknown true graph G. We regard that a deterministic algorithm $\mathcal{A}$ is stable when the Hamming distance $d_{\mathrm{Ham}}\big(\mathcal{A}(G), \mathcal{A}(G')\big)$ is small, where $\mathcal{A}(G)$ and $\mathcal{A}(G')$ are outputs of $\mathcal{A}$ on G and G', respectively. Here, outputs are typically vertex sets or edges sets. More specifically, for an integer $k \geq 1$ and a function β on graphs, we say that the *k-average sensitivity* of a deterministic algorithm $\mathcal{A}$ is at most β if

$$\mathop{\mathbb{E}}_{e_1,\ldots,e_k \sim E}\big[d_{\mathrm{Ham}}\big(\mathcal{A}(G), \mathcal{A}(G - \{e_1, \ldots, e_k\})\big)\big] \leq \beta(G) \tag{1}$$

for every graph $G = (V, E)$, where $G - F$ for an edge set F is the subgraph obtained from G by removing F, and $e_1, \ldots, e_k$ are sampled from E uniformly at random. When $k = 1$, we say that the *average sensitivity* is at most β. Informally, we say that algorithms with low (k-)average sensitivity are *averagely stable*. 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 $G = (V, E)$, 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 $\mathcal{A}$, 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 $\mathcal{A}$ 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 define average sensitivity of randomized algorithms. Abusing the notation, we regard $\mathcal{A}(G)$ as the distribution of the output of $\mathcal{A}$ on G, and let $d_{\mathrm{EM}}(\mathcal{A}(G), \mathcal{A}(G'))$ 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 $k \geq 1$ and a function β on graphs, we say that the *k-average sensitivity* of a randomized algorithm $\mathcal{A}$ is at most β if

$$\mathop{\mathbb{E}}_{e_1,\ldots,e_k \sim E} \left[d_{\mathrm{EM}}\big(\mathcal{A}(G), \mathcal{A}(G - \{e_1, \ldots, e_k\})\big) \right] \leq \beta(G), \tag{2}$$

where $e_1, \ldots, e_k$ are sampled from E uniformly at random. When $k = 1$, again, we say that the *average sensitivity* is at most β. Note that when the algorithm $\mathcal{A}$ is deterministic, (2) matches the definition of 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 $\mathbb{E}_{e_1,\ldots,e_k \sim E} \left[d_{\mathrm{TV}}\big(\mathcal{A}(G), \mathcal{A}(G - \{e_1, \ldots, e_k\})\big) \right]$, where $d_{\mathrm{TV}}(\cdot, \cdot)$ 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 $\mathsf{H} \cdot \gamma(G)$, where the H is the maximum Hamming weight of a solution.

Example 1.4. Randomness does not add any power to 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 lends itself to 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 averagely stable 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.

k-average sensitivity from 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 removal of a single edge automatically gives a bound on the average sensitivity of that algorithm with respect to 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 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 composability. It will be useful if we can sequentially apply averagely stable algorithms on the input to get a solution and the whole algorithm is again averagely stable. We show two different sequential composition theorems for average sensitivity.

Theorem 1.6 (Sequential composability)**.** *Consider two randomized algorithms $\mathcal{A}_1 : \mathcal{G} \to \mathcal{S}_1, \mathcal{A}_2 : \mathcal{G} \times \mathcal{S}_1 \to \mathcal{S}_2$. Suppose that the average sensitivity of $\mathcal{A}_1$ with respect to the total variation distance is γ_1 and the average sensitivity of $\mathcal{A}_2(\cdot, S_1)$ is $\beta_2^{(S_1)}$ for any $S_1 \in \mathcal{S}_1$. Let $\mathcal{A} : \mathcal{G} \to \mathcal{S}_2$ be a randomized algorithm obtained by composing $\mathcal{A}_1$ and $\mathcal{A}_2$, that is, $\mathcal{A}(G) = \mathcal{A}_2(G, \mathcal{A}_1(G))$. Then, the average sensitivity of $\mathcal{A}$ is $\mathsf{H} \cdot \gamma_1(G) + \mathbb{E}_{S_1 \sim \mathcal{A}_1(G)}\left[\beta_2^{(S_1)}(G)\right]$, where H denotes the maximum Hamming weight among those of solutions obtained by running $\mathcal{A}$ on G and $\{G - e\}_{e \in E}$.*

Our second composition theorem is for the average sensitivity with respect to the total variation distance. This is also useful to analyze 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 composability w.r.t. the TV distance)**.** *Consider k randomized algorithms $\mathcal{A}_i : \mathcal{G} \times \prod_{j=1}^{i-1} \mathcal{S}_j \to \mathcal{S}_i$ for $i \in \{1, \ldots, k\}$. Suppose that, for each $i \in \{1, \ldots, k\}$, the average sensitivity of $\mathcal{A}_i(\cdot, S_1, \ldots, S_{i-1})$ is γ_i with respect to the total variation distance for every $S_1 \in \mathcal{S}_1, \ldots, S_{i-1} \in \mathcal{S}_{i-1}$. Consider a sequence of computations $S_1 = \mathcal{A}_1(G), S_2 = \mathcal{A}_2(G, S_1), \ldots, S_k = \mathcal{A}_k(G, S_1, \ldots, S_{k-1})$. Let $\mathcal{A} : \mathcal{G} \to \mathcal{S}_k$ be a randomized algorithm that performs this sequence of computations on input G and outputs S_k. Then, the average sensitivity of $\mathcal{A}$ is at most $\sum_{i=1}^{k} \gamma_i(G)$ with respect to the total variation distance.*

Parallel composability. It is often the case that there are multiple algorithms that solve the same problem albeit with different average sensitivity guarantees. Such averagely stable 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 the component algorithms for all graphs.

Theorem 1.8 (Parallel composability)**.** *Let $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_k$ be algorithms for a graph problem with average sensitivities $\beta_1, \beta_2, \ldots, \beta_k$, respectively. Let $\mathcal{A}$ be an algorithm that, given a graph G, runs $\mathcal{A}_i$ 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 $\mathcal{A}$ on G and $\{G - e\}_{e \in E}$. Then the average sensitivity of $\mathcal{A}$ is at most $\sum_{i \in [k]} \rho_i(G) \cdot \beta_i(G) + \mathsf{H} \cdot \mathbb{E}_{e \in E}\left[\sum_{i \in [k]} |\rho_i(G) - \rho_i(G - e)|\right]$.*

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 a global algorithm and the query complexity of a local algorithm that simulates oracle access to the output of the global algorithm. Roughly speaking, we show, in Theorem 1.9, that the existence of a local algorithm $\mathcal{O}$ that can answer queries about the solution produced by a global algorithm $\mathcal{A}$ implies that the average sensitivity of $\mathcal{A}$ is bounded by the query complexity of $\mathcal{O}$. We use Theorem 1.9 to prove the existence of averagely stable matching algorithms based on the sublinear-time matching algorithms due to Yoshida et al. [YYI12].

Theorem 1.9 (Locality implies low average sensitivity)**.** *Consider a randomized algorithm $\mathcal{A} : \mathcal{G} \to \mathcal{S}$ for a graph problem, where the solutions are subsets of the set of edges of the input graph. Assume that there exists an oracle $\mathcal{O}$ satisfying the following:*

Table 1: Our results. Here n, m, OPT denote the number of vertices, the number of edges, and the optimal value, respectively, and $\varepsilon \in (0,1)$ is an arbitrary constant. The notation $\widetilde{O}(\cdot)$ hides a polylogarithmic factor in n. Small additive losses in the approximation guarantees are omitted.

Problem	Output	Approximation Ratio	Average Sensitivity	Reference
Minimum Spanning Forest	Edge set	1	$O\left(\frac{n}{m}\right)$	Section 2
Global Minimum Cut	Vertex set	$2+\varepsilon$	$n^{O(1/\varepsilon \mathsf{OPT})}$	Section 3
Maximum Matching	Edge set	$1/2 - o(1)$	$\widetilde{O}(\mathsf{OPT}^{3/4})$	Section 4.2
		$1-\varepsilon$	$\widetilde{O}\left(\left(\mathsf{OPT}/\varepsilon^3\right)^{1/(1+\Omega(\varepsilon^2))}\right)$	Section 4.3
Minimum Vertex Cover	Vertex set	$2+o(1)$	$\widetilde{O}(\mathsf{OPT}^{3/4})$	Section 5.1
		$\widetilde{O}\left(\frac{m^{1+\varepsilon}}{n}\right)$	$O\left(\frac{n^2}{m^{1+\varepsilon}}\right)$	Section 5.2
2-Coloring	Vertex set	—	$\Omega(n)$	Section 6

- *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 $\mathcal{A}$ on G with π as its random string,*
- *the oracle $\mathcal{O}$ makes at most $q(G)$ queries to G in expectation, where this expectation is taken over the random coins of $\mathcal{A}$ and a uniformly random query $e \in E$.*

Then, $\mathcal{A}$ has average sensitivity at most $q(G)$. Moreover, this is also true for algorithms for graph problems, where the solutions are subsets of the set of vertices of the input graph, whenever $|E| \geq |V|$.

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. As an indirect method to bound 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 Averagely stable 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, 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, $O(\mathsf{OPT})$ 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 [Kru56] 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 OPT, and hence we want to make its average sensitivity smaller than OPT. We observe that the average sensitivity becomes smaller than OPT when $\mathsf{OPT} = \Omega(k \log\log k / \log k)$ for $k = \log(n)/\varepsilon$, and it quickly decreases as OPT increases.

For the maximum matching problem, we propose two algorithms. The first one has approximation ratio $1/2 - o(1)$ and average sensitivity $\widetilde{O}(\mathsf{OPT}^{3/4})$, which is much smaller than the trivial $O(\mathsf{OPT})$. The second one has approximation ratio $1-\varepsilon$ and average sensitivity $\widetilde{O}\left(\left(\mathsf{OPT}/\varepsilon^3\right)^{1/(1+\Omega(\varepsilon^2))}\right)$ for every constant $\varepsilon \in (0,1)$, which shows that we do not have to sacrifice the approximation ratio a lot to obtain a non-trivial average sensitivity.

For the minimum vertex cover problem, we propose two algorithms. The first algorithm has approximation ratio $2 + o(1)$, which is close to the best we can hope for as obtaining $(2-\varepsilon)$-approximation is NP-Hard assuming the Unique Games conjecture [KR03]. Moreover, the average sensitivity of $\widetilde{O}(\mathsf{OPT}^{3/4})$ is much smaller than the trivial $O(\mathsf{OPT})$. The second algorithm has a worse approximation ratio but can achieve a better average sensitivity in some regimes. For example, when $\mathsf{OPT} = \Omega(n)$, $m = \Theta(n)$ and $\varepsilon = 1/2$, the average sensitivity of the first algorithm is $\Omega(n^{3/4})$ whereas that of the second algorithm is $O(n^{1/2})$.

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 $\Omega(n)$ in the average sensitivity, that is, there is no algorithm with non-trivial average sensitivity.

1.4 Discussions

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 OPT could be much smaller than H, such a normalization can diminish subtle differences in average sensitivity, e.g., $O(\mathsf{OPT}^{1/2})$ vs $O(\mathsf{OPT})$. 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 [MSVW18]. However, in our context, it seems 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 for most applications.

Average sensitivity against 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 [ER59], which does not well-represent real networks such as social networks and road networks. 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 $\mathcal{A}$ 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 $\mathcal{A}_\pi$ on G, where the algorithm $\mathcal{A}_\pi$ has π hardwired into it. Let us assume that $\mathcal{A}_\pi$ can be applied to any graph. It is also natural to define the average sensitivity of $\mathcal{A}$ as

$$\mathop{\mathbb{E}}_{e\sim E}\left[\mathop{\mathbb{E}}_{\pi}\left[d_{\mathrm{Ham}}\big(\mathcal{A}_\pi(G), \mathcal{A}_\pi(G-e)\big)\right]\right]. \tag{3}$$

In other words, we measure the expected distance between the outputs of $\mathcal{A}$ on G and $G-e$ when we feed the same string π to $\mathcal{A}$, over the choice of π 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 $\mathcal{A}(G)$ to $\mathcal{A}(G-e)$ 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_π can be applied to any graph does not hold in general, and bounding (3) is unnecessarily tedious and is not very enlightening.

1.5 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 [Bav50, Bea65, Sab66], harmonic centrality [ML00], betweenness centrality [Fre77], and PageRank [PBMW99]. To compare these centralities qualitatively, Murai and Yoshida [MY19] 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

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

where e and v are sampled uniformly at random. They showed various upper and lower bounds for centralities. See [MY19] 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* [DMNS06] is a notion closely related to average sensitivity. It considers a neighbor relation over inputs and asks 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. [NRS07] and further studied by [HLMJ09, GLM+10, KS12, KNRS13, KRSY14, RS16]. Here, the neighbors of a graph $G = (V, E)$ are defined to be $\{G-e\}_{e\in E}$. Then for $\varepsilon > 0$, we say that an algorithm is *ε-differentially private* if for all $e \in E$,

$$\exp(-\varepsilon) \cdot \Pr[\mathcal{A}(G - e) \in \mathcal{S}] \leq \Pr[\mathcal{A}(G) \in \mathcal{S}] \leq \exp(\varepsilon) \cdot \Pr[\mathcal{A}(G - e) \in \mathcal{S}] \tag{4}$$

for any set of solutions $\mathcal{S}$.

As differential privacy imposes the constraint (4) for every $e \in E$, the requirement is sometimes too strong for graph problems. 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 $n - 1$. To avoid this issue, Gupta et al. [GLM+10] considered 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 $\mathcal{A}(G)$ and $\mathcal{A}(G - e)$ 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. We would like to add that, despite these differences, our algorithms for the global minimum cut problem and the vertex cover problem are inspired by differentially private algorithms for the same problems [GLM+10].

Generalization and stability of learning algorithms. Generalization [SSBD09] is a fundamental concept in statistical learning theory. Given samples $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n$ from an unknown true distribution $\mathcal{D}$ over a dataset, the goal of a learning algorithm $\mathcal{L}$ is to output a parameter θ that minimizes *expected loss* $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{D}}[\ell(\boldsymbol{z}; \theta)]$, where $\ell(\boldsymbol{z}; \theta)$ is the loss incurred by a sample $\boldsymbol{z}$ with respect to a parameter θ. As the true distribution $\mathcal{D}$ is unknown, a frequently used approach in learning is to compute a parameter θ that minimizes the *empirical loss* $\frac{1}{n} \cdot \sum_{i=1}^{n} \ell(\boldsymbol{z}_i; \theta)$, which is an unbiased estimator of the expected loss and is purely a function of the available samples. The *generalization error* of a learner $\mathcal{L}$ 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 [BE02]. This also ensures that the learned parameter θ 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 [BE02] defined a notion of stability for learning algorithms and explored its connection to 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, average sensitivity considers removing *random* edges from a graph. Also, average sensitivity considers the change of the output solution rather than that of the objective value.

1.6 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; it is always 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 rule of breaking ties among edges.

Global minimum cut. For the global minimum cut problem, our algorithm is inspired by a differentially private algorithm due to Gupta et al. [GLM+10]. Our algorithm, given a parameter $\varepsilon > 0$ and a graph G as input, first enumerates a list of cuts whose sizes are at most $(2+\varepsilon) \cdot \mathsf{OPT}$; this enumeration can be done in polynomial time as shown by Karger's theorem [Kar93]. It then outputs a cut from the list with probability inversely proportional to the exponential of the product of the size of the cut and $O(1/\varepsilon\mathsf{OPT})$. 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 $n^{O(1/\varepsilon\mathsf{OPT})}$.

Maximum matching. There are several components to the design and analysis of our averagely stable $(\frac{1}{2}-\varepsilon)$-approximation algorithm for the maximum matching problem. Our starting point is the observation (Theorem 1.9) that the ability to locally simulate access to the solution of an algorithm $\mathcal{A}$ implies that $\mathcal{A}$ is averagely stable. We use this to bound the average sensitivity of a randomized greedy $\frac{1}{2}$-approximation algorithm $\mathcal{A}$ for the maximum matching problem. Specifically, $\mathcal{A}$ constructs a maximal matching by iterating over edges in the input graph $G = (V, E)$ according to a uniformly random ordering and adding an edge to the current matching if the addition does not violate the matching property. Yoshida et al. [YYI12] constructed a local algorithm that, given a uniformly random edge $e \in E$ as input, makes $O(\Delta)$ queries to G in expectation and answers whether e is in the matching output by $\mathcal{A}$ on G, where the expectation is over the choice of input e and the randomness in $\mathcal{A}$, and Δ is the maximum degree of G. Combined with Theorem 1.9, this implies that the average sensitivity of $\mathcal{A}$ is $O(\Delta)$.

Next, we transform $\mathcal{A}$ to also work for graphs of unbounded degree as follows. The idea is to remove vertices of degree at least $\frac{m}{\varepsilon\mathsf{OPT}}$ from the graph and run $\mathcal{A}$ on the resulting graph. This transformation affects the approximation guarantee only by an additive $\varepsilon\mathsf{OPT}$ 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 of G and $G - e$ are different for any $e \in E$.

We circumvent this issue by using a Laplace random variable L as the threshold, where the distribution of L is tightly concentrated around $\frac{m}{\varepsilon\mathsf{OPT}}$. 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 $\mathcal{A}_x$ that, on input a 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 $\mathcal{O}$ with query complexity $q(\Delta)$ that simulates oracle access to the solutions output by an algorithm $\mathcal{A}$, we can, for all $x \in \mathbb{R}$, construct a local algorithm $\mathcal{O}_x$ for the algorithm $\mathcal{A}_x$. Moreover, the query complexity of $\mathcal{O}_x$, which also bounds the average sensitivity of $\mathcal{A}_x$ by Theorem 1.9, is at most $O(x^2 q(x))$. This implies that the second term in the expression

given by Theorem 1.6, which is given by $\mathbb{E}_L\left[O(L^2 q(L))\right]$, is $O((\frac{m}{\varepsilon \mathsf{OPT}})^3)$.

An issue with the aforementioned matching algorithm is that its average sensitivity is poor for graphs with small values of OPT. However, we observe that the algorithm that simply outputs the lexicographically smallest maximum matching does not have this issue. Its average sensitivity is $O(\mathsf{OPT}^2/m)$, since the output matching stays the same unless an edge in the matching is removed. We obtain our final averagely stable $(\frac{1}{2}-\varepsilon)$-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 sensitivity of the resultant algorithm as $O\left((\mathsf{OPT}/\varepsilon)^{3/4}\right)$.

The design and analysis of our averagely stable $(1-\varepsilon)$-approximation algorithm for the maximum matching problem uses similar ideas as above. The only difference is that we replace the randomized greedy maximal matching algorithm above with a $(1-\varepsilon)$-approximation algorithm that repeatedly improves a matching using greedily chosen augmenting paths.

Minimum vertex cover. We describe two averagely stable algorithms for the minimum vertex cover problem. Our $(2+\varepsilon)$-approximation algorithm is based on a reduction from the averagely stable $(\frac{1}{2}-\varepsilon)$-approximation algorithm for the maximum matching problem. In particular, it runs the averagely stable matching algorithm and outputs a union of the set of vertices removed (by thresholding) and the set of endpoints of the matching computed. For the approximation guarantee, we argue that, with high probability, the cardinality of the set of removed vertices is $O(\varepsilon\mathsf{OPT})$. The main task in showing that the algorithm is averagely stable is to bound the average sensitivity of outputting the set of removed vertices. In case the same value of threshold is used for G and $G-e$, the cardinality of symmetric difference between the sets of removed vertices is at most 2. Using this observation and the ideas used in bounding the average sensitivity of our matching algorithms, we can bound the average sensitivity of outputting the set of removed vertices.

Our second algorithm for vertex cover is based on a differentially private vertex cover approximation algorithm due to Gupta et al. [GLM+10]. Specifically, we output a permutation of the vertices and for each edge, its first endpoint in the permutation is in the vertex cover. If we generate our permutation by repeatedly sampling vertices according to their yet *uncovered* degree, we get a 2-approximation algorithm for vertex cover [Pit85]. If we instead output a uniformly random permutation of vertices, we get an algorithm with good average sensitivity but poor approximation guarantee. Our algorithm finds a middle ground between these approaches, by selecting vertices with probability proportional to their *uncovered* degrees in the beginning and progressively skewing towards the uniform distribution.

2-coloring. To show our $\Omega(n)$ lower bound of 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-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-path are neighbors if the latter is obtained from the former by removing an edge. A 2-path has at most four neighbors. The output distribution of any 2-coloring algorithm $\mathcal{A}$ on a 2-path can be close (in earth mover's distance) only to those of at most 2 of its neighboring paths. If $\mathcal{A}$ however has low average sensitivity, the output distributions of $\mathcal{A}$ has to be close on a large fraction of pairs of neighboring graphs, which gives a contradiction.

1.7 Notation

For a positive integer n, let $[n] = \{1, 2, \ldots, n\}$. Let $G = (V, E)$ be a graph. 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 $G[S]$ be the subgraph of G induced by S. We often use the symbols n, m, Δ to denote the number of vertices, the number of edges, and the maximum degree of a vertex, respectively, in the input graph. We use $\mathsf{OPT}(G)$ to denote the optimal value of a graph G in the graph problem we are concerned with. We simply write OPT when G is clear from the context. We denote by $\mathcal{G}$ the (infinite) set consisting of all graphs.

1.8 Organization

We show our averagely stable algorithms for the minimum spanning forest problem, the global minimum cut problem, the maximum matching problem, and the vertex cover problems in Sections 2, 3, 4, and 5, respectively. Then, we show a linear lower bound for the 2-coloring problem in Section 6. We discuss general properties of average sensitivity in Section 7.

2 Warm Up: Minimum Spanning Forest

To get intution 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 $w : E \to \mathbb{R}$ 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 [Kru56] 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 averagely stable.

Theorem 2.1. *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

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

In Appendix A, we show that Prim's algorithm, another classical algorithm for the minimum spanning forest problem, have average sensitivity $\Omega(m)$ for a certain natural tie breaking rule.

3 Global Minimum Cut

For a graph $G = (V, E)$ and a vertex set $S \subseteq V$, we define $\mathsf{cost}(G, S)$ 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 $\emptyset \subsetneq S \subsetneq V$ that minimizes $\mathsf{cost}(G, S)$. In this section, we show an algorithm with low average sensitivity for computing the global minimum cut problem in undirected graphs. Specifically, we show the following.

Theorem 3.1. *For $\varepsilon > 0$, there exists a polynomial-time algorithm for the global minimum cut problem with approximation ratio $2 + \varepsilon$ and average sensitivity $n^{O(1/\varepsilon \mathsf{OPT})}$.*

Let 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 $\exp(-\alpha \cdot \mathsf{cost}(G, S))$ for a suitable α. See Algorithm 1 for details.

Algorithm 1: Stable Algorithm for Global Minimum Cut

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

1 Compute the value OPT;

2 Let $\alpha \leftarrow \frac{(2+1/\varepsilon)\log n}{\mathsf{OPT}}$ denote a parameter;

3 Enumerate all cuts of size at most $(2 + 7\varepsilon)\mathsf{OPT} + 2\varepsilon$;

4 Sample a vertex set S (from among the cuts enumerated) with probability proportional to $\exp(-\alpha \cdot \mathsf{cost}(G, S))$;

5 **return** S.

The approximation ratio of the Algorithm 1 is $2 + 9\varepsilon$: It clearly holds when $\mathsf{OPT} \geq 1$, and it also holds when $\mathsf{OPT} = 0$ because we only output a cut of size zero (for $\varepsilon < 1/2$). The following theorem due to Karger [Kar93] directly implies that it runs in time polynomial in the input size for any constant $\varepsilon > 0$.

Theorem 3.2 ([Kar93])**.** *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 averagely stable.

Lemma 3.3. *The average sensitivity of Algorithm 1 is at most*

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

As we have $\mathsf{OPT} \leq 2m/n$, the average sensitivity can be bounded by $n^{O(1/\varepsilon \mathsf{OPT})}$, and Theorem 3.1 follows by replacing ε with $\varepsilon/9$.

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

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 $\exp(-\alpha \cdot \mathsf{cost}(G,S))$. 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

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

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

Now, we start with $\mathcal{A}'(G)$. For each $S \subseteq V$ such that $\mathsf{cost}(G,S) \leq \mathsf{OPT} + b$, 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 $n \cdot (p'_G(S) - \frac{Z}{Z'} \cdot p'_G(S))$. For each $S \subseteq V$ such that $\mathsf{cost}(G,S) > \mathsf{OPT} + b$, we move a mass of $p'_G(S)$ at a cost of $n \cdot p'_G(S)$. The total cost of moving masses is then equal to:

$$\begin{aligned}
d_{\mathrm{EM}}\left(\mathcal{A}(G), \mathcal{A}'(G)\right) &\leq n \cdot \sum_{S \subseteq V : \mathsf{cost}(G,S) \leq \mathsf{OPT}+b} p'_G(S)\left(1 - \frac{Z}{Z'}\right) + n \cdot \sum_{S \subseteq V : \mathsf{cost}(G,S) > \mathsf{OPT}+b} p'_G(S) \\
&= \frac{n(Z'-Z)}{Z'}\left(\sum_{S \subseteq V : \mathsf{cost}(G,S) \leq \mathsf{OPT}+b} p'_G(S)\left(1 - \frac{Z}{Z'}\right) + 1\right) \\
&\leq \frac{2n(Z'-Z)}{Z'}.
\end{aligned}$$

Let n_t stand for the number of cuts of cost at most $\mathsf{OPT} + t$ in G. By Karger's theorem (Theorem 3.2), we have that $n_t \leq n^{2+2t/\mathsf{OPT}}$. Then, we have

$$\begin{aligned}
\frac{Z'-Z}{Z'} &\leq \sum_{t>b} \exp(-\alpha t) \cdot (n_t - n_{t-1}) \leq (\exp(\alpha) - 1) \cdot \sum_{t>b} \exp(-\alpha t) n_t \\
&\leq (\exp(\alpha) - 1) n^2 \cdot \sum_{t>b} n^{2t/\mathsf{OPT}} \cdot \exp(-\alpha t) \\
&\leq (\exp(\alpha) - 1) n^2 \cdot \sum_{t>b} n^{-t/\varepsilon\mathsf{OPT}} \leq (\exp(\alpha) - 1) n^2 \cdot \frac{n^{-(b+1)/\varepsilon\mathsf{OPT}}}{1 - n^{-1/\varepsilon\mathsf{OPT}}} \\
&= \left(n^{(2+1/\varepsilon)/\mathsf{OPT}} - 1\right) \cdot \left(1 + \frac{1}{n^{1/\varepsilon\mathsf{OPT}} - 1}\right) \cdot \frac{n^2}{n^{(b+1)/\varepsilon\mathsf{OPT}}} \\
&\leq n^{(2+1/\varepsilon)/\mathsf{OPT}} \cdot \left(1 + \frac{\varepsilon n}{\log n}\right) \cdot \frac{n^2}{n^{(b+1)/\varepsilon\mathsf{OPT}}} \\
&= O\left(\frac{\varepsilon n^{3+(2+1/\varepsilon)/\mathsf{OPT}}}{n^{(b+1)/\varepsilon\mathsf{OPT}}}\right) = O\left(\frac{\varepsilon}{n^{4+1/\varepsilon}}\right).
\end{aligned}$$

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 $d_{\mathrm{EM}}\left(\mathcal{A}(G), \mathcal{A}'(G)\right) \leq O(\frac{\varepsilon}{n^{3+1/\varepsilon}})$. In addition, we can bound the expected size of the cut output by $\mathcal{A}'$ on G as $\mathsf{OPT} + b + m \cdot O(\frac{\varepsilon}{n^{3+1/\varepsilon}}) = (2+7\varepsilon)\mathsf{OPT} + 2\varepsilon + O(\frac{\varepsilon m}{n^{4+1/\varepsilon}})$.

We now bound the earth mover' distance between $\mathcal{A}'(G)$ and $\mathcal{A}'(G-e)$ for an arbitrary edge $e \in E$. Let Z'_e denote the quantity $\sum_{S \subseteq V} \exp(-\alpha \cdot \mathsf{cost}(G-e,S))$. 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,

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

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,

$$d_{\mathrm{EM}}\left(\mathcal{A}'(G), \mathcal{A}'(G-e)\right) \leq n \cdot \left(\frac{Z'_e}{Z'} - 1\right) \cdot \sum_{S \subseteq V} p'_G(S) = n \cdot \left(\frac{Z'_e}{Z'} - 1\right).$$

By the triangle inequality, the earth mover's distance between $\mathcal{A}(G)$ and $\mathcal{A}(G-e)$ can be bounded as

$$\begin{aligned} d_{\mathrm{EM}}\big(\mathcal{A}(G), \mathcal{A}(G-e)\big) &\leq d_{\mathrm{EM}}\left(\mathcal{A}(G), \mathcal{A}'(G)\right) + d_{\mathrm{EM}}\left(\mathcal{A}'(G), \mathcal{A}'(G-e)\right) + d_{\mathrm{EM}}\left(\mathcal{A}'(G-e), \mathcal{A}(G-e)\right) \\ &\leq n \cdot \left(\frac{Z'_e}{Z'} - 1\right) + O\left(\frac{2\varepsilon}{n^{2+1/\varepsilon}}\right). \end{aligned}$$

Hence, the average sensitivity of $\mathcal{A}$ is bounded as:

$$\begin{aligned} \beta(G) = \mathop{\mathbb{E}}_{e \in E} d_{\mathrm{EM}}\big(\mathcal{A}(G), \mathcal{A}(G-e)\big) &\leq O\left(\frac{2\varepsilon}{n^{3+1/\varepsilon}}\right) + n \cdot \mathop{\mathbb{E}}_{e \in E}\left(\frac{Z'_e}{Z'} - 1\right) \\ &= O\left(\frac{2\varepsilon}{n^{3+1/\varepsilon}}\right) + \frac{n}{mZ'} \sum_{e \in E} (Z'_e - Z') \\ &= O\left(\frac{2\varepsilon}{n^{3+1/\varepsilon}}\right) + \frac{n}{mZ'} \sum_{e \in E} \sum_{S \subseteq V : e \text{ crosses } S} \exp(-\alpha \cdot \mathsf{cost}(G-e,S)) - \exp(-\alpha \cdot \mathsf{cost}(G,S)) \\ &= O\left(\frac{2\varepsilon}{n^{3+1/\varepsilon}}\right) + \frac{n(\exp(\alpha)-1)}{mZ'} \sum_{e \in E} \sum_{S \subseteq V : e \text{ crosses } S} \exp(-\alpha \cdot \mathsf{cost}(G,S)) \\ &= O\left(\frac{2\varepsilon}{n^{3+1/\varepsilon}}\right) + \frac{n(\exp(\alpha)-1)}{m} \sum_{S \subseteq V} \mathsf{cost}(G,S) \cdot \frac{\exp(-\alpha \cdot \mathsf{cost}(G,S))}{Z'}. \end{aligned}$$

The summation in the second term above is equal to the expected size of the cut output by algorithm $\mathcal{A}'$ on input G. We argued that it is at most $(2+7\varepsilon)\mathsf{OPT} + 2\varepsilon + O(\frac{\varepsilon m}{n^{4+1/\varepsilon}})$. Hence, the average sensitivity of $\mathcal{A}$ is at most

$$\begin{aligned} &\frac{n}{m} \cdot n^{(2+1/\varepsilon)/\mathsf{OPT}} \cdot ((2+7\varepsilon)\,\mathsf{OPT} + 2\varepsilon) + O\left(\frac{\varepsilon n^{(2+1/\varepsilon)/\mathsf{OPT}} + 2}{n^{3+1/\varepsilon}}\right) \\ &= \frac{n}{m} \cdot n^{(2+1/\varepsilon)/\mathsf{OPT}} \cdot ((2+7\varepsilon)\,\mathsf{OPT} + 2\varepsilon) + o(1) \end{aligned}$$

as $\mathsf{OPT} \geq 1$. □

4 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 different algorithms with low average sensitivity that approximate the maximum matching in a graph.

4.1 Lexicographically smallest matching

In this section, we describe an algorithm that computes a maximum matching in a graph with average sensitivity at most OPT^2/m and prove Theorem 4.1, where 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 [Edm65].

Theorem 4.1. *Let $\mathcal{A}$ be the algorithm that outputs the lexicographically smallest maximum matching. Then, the average sensitivity of $\mathcal{A}$ is at most OPT^2/m, where 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

$$\frac{\mathsf{OPT}}{m} \cdot \mathsf{OPT} + \left(1 - \frac{\mathsf{OPT}}{m}\right) \cdot 0 = \frac{\mathsf{OPT}^2}{m}. \qquad \square$$

Remark 4.2. Consider the path graph $P_n = (\{1, \dots, n\}, E)$, where $E = \{(i, i+1) : i \in [n-1]\}$. The average sensitivity of the above algorithm on P_n is $\Omega(\frac{\mathsf{OPT}^2}{m})$. Hence the above analysis of the average sensitivity is tight.

4.2 Greedy matching algorithm

In this section, we describe an algorithm (based on a randomized greedy maximal matching algorithm) with average sensitivity $\tilde{O}\left(\mathsf{OPT}^{3/4}\right)$ and approximation ration $1/2 - o(1)$ for the maximum matching problem and prove Theorem 4.9.

In Theorem 4.3, we prove that the basic randomized greedy maximal matching algorithm has sensitivity $O(\Delta)$, where Δ is the maximum degree of the input graph.

Theorem 4.4 shows how to transform the randomized greedy algorithm to another algorithm whose average sensitivity does not depend on the maximum degree, albeit at the cost of slightly worsening the approximation guarantee. In particular, Theorem 4.8 shows that a $(1/2 - \varepsilon)$-approximation algorithm for maximum matching with average sensitivity $O\left(\frac{\varepsilon}{1-\varepsilon} \log n + \frac{m^3}{\varepsilon^3 \mathsf{OPT}^3}\right)$ is obtained by applying Theorem 4.4 to the randomized greedy maximal matching.

Finally, we combine the matching algorithm guaranteed by Theorem 4.8 with the matching algorithm guaranteed by Theorem 4.1 using the parallel composition property (Theorem 7.2) of averagely stable algorithms and obtain Theorem 4.9.

4.2.1 Average sensitivity of the greedy algorithm in terms of the maximum degree

In this section, we describe the average sensitivity guarantee of the randomized greedy algorithm described in Algorithm 2. It is evident that Algorithm 2 runs in polynomial time and that the

Algorithm 2: Randomized Greedy Algorithm

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

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

matching it outputs has size at least $\frac{1}{2}$ the size of a maximum matching in the input graph.

Theorem 4.3. *For every undirected unweighted graph G, the average sensitivity of Algorithm 2 is $\beta(G) \leq \frac{1}{2} + \Delta$, where Δ is the maximum degree of G.*

Proof. Consider a graph $G = (V, E)$. Let Δ be the maximum degree of G. For an edge $e \in E$, let $G - e$ denote the graph obtained by removing e from G. Let $M(G)$ denote the matching output by Algorithm 2 on input G. Yoshida et al. [YYI12, Theorem 2.1] show that the presence of a uniformly random edge e in $M(G)$ depends on at most $\frac{1}{2} + \Delta$ edges in expectation, where the expectation is taken over both the randomness of the algorithm and the randomness in selecting the edge e. By applying Theorem 1.9 to this statement, we can see that the average sensitivity of Algorithm 2 is $\beta(G) \leq \frac{1}{2} + \Delta$, where Δ is the maximum degree of G. □

4.2.2 Averagely stable 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 averagely stable algorithms for graphs with unbounded degree. Let $\mathsf{Lap}(\mu, \phi)$ denote the Laplace distribution with a location parameter μ and a scale parameter ϕ.

Theorem 4.4. *Let $\mathcal{A}'$ be a randomized algorithm for the maximum matching problem such that the size of the matching output by $\mathcal{A}'$ on a graph G is always at least $a \cdot \mathsf{OPT}$ for some $a \geq 0$. In addition, assume that there exists an oracle $\mathcal{O}$ 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 $\mathcal{A}'$ on G with π as its random string, and*
- *the oracle $\mathcal{O}$ makes at most $q(\Delta)$ queries to G in expectation, where Δ is the maximum degree of G and the expectation is taken over the random coins of $\mathcal{A}'$ and a uniformly random query $e \in E$.*

Let $\delta > 0$ and τ be a non-negative function on graphs. Then, there exists an algorithm $\mathcal{A}$ for the maximum matching problem with average sensitivity

$$\beta(G) \leq O\left(\frac{K_G}{\delta(\tau(G) - K_G)} + \exp\left(-\frac{1}{\delta}\right)\right) \cdot \mathsf{OPT} + \mathop{\mathbb{E}}_{L}\left[(2L-2)^2 q(L)\right],$$

where L is a random variable distributed as $\mathsf{Lap}(\tau(G), \delta\tau(G))$ and $K_G = \max_{e \in E(G)} |\tau(G) - \tau(G-e)|$. Moreover, the expected size of the matching output by $\mathcal{A}$ is at least

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

The following fact will be useful in the proof of Theorem 4.4.

Proposition 4.5. *Let L be a random variable distributed as $\mathsf{Lap}(\mu, \phi)$. 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 4.4. The algorithm $\mathcal{A}$ is given below.

Algorithm $\mathcal{A}$: On input $G = (V, E)$,

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

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

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

Claim 4.6. *For $x \in \mathbb{R}$, $\mathbb{E}_{e \in E}\left[d_{\mathrm{EM}}\big(\mathcal{A}'([G]_x), \mathcal{A}'([G-e]_x)\big)\right] \leq (2x-2)^2 q(x)$.*

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

In order to bound the average sensitivity of $\mathcal{A}'_x$, construct an oracle $\mathcal{O}_x$ as follows. $\mathcal{O}_x$ 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 $\mathcal{A}'_x$ on G. Otherwise, it runs $\mathcal{O}$ with access to $[G]_x$ and e as input and outputs the answer of $\mathcal{O}$.

We can analyze the query complexity of $\mathcal{O}_x$ 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 $\mathcal{O}_x$ can check whether an edge $e = (u, v)$ is alive or not by querying at most $2x - 2$ edges incident to e. In particular $\mathcal{O}_x$ examines the neighbors of u and v one by one, and, as soon

$\mathcal{O}_x$ encounters $x-1$ distinct neighbors (excluding u or v themselves) for either u or v, $\mathcal{O}_x$ can declare e to be a dead edge.

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

If the input edge $e \in E$ is alive, then we know that it is a uniformly random alive edge. By the guarantee on $\mathcal{O}$, we then know that $\mathcal{O}$ makes at most $q(x)$ queries to the alive edges in expectation over the randomness of $\mathcal{A}'$ and the choice of the input alive edge, since the maximum degree of $[G]_x$ is at most x. In order for the oracle $\mathcal{O}_x$ to simulate oracle access to $[G]_x$ for the oracle $\mathcal{O}$, for each alive edge e queried by $\mathcal{O}$, the oracle $\mathcal{O}_x$ 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, $\mathcal{O}_x$ 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 $\mathcal{O}_x$ is at most $(2x-2)^2 q(x)$, where the expectation is taken over the edges of $e \in E$ and the randomness in $\mathcal{A}_x$.

Therefore, by Theorem 1.9, we get that the average sensitivity of algorithm $\mathcal{A}_x$ is bounded by $(2x-2)^2 q(x)$. □

We now bound the quantity $\mathbb{E}_{e\in E}\left[d_{\mathrm{TV}}(L, L_e)\right]$.

Claim 4.7. *For any $e \in E$, we have*

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

Proof. Let $f_L, f_{L_e} \colon \mathbb{R} \to \mathbb{R}$ 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

$$\begin{aligned}\frac{f_L(x)}{f_{L_e}(x)} &= \frac{\frac{1}{2\delta\tau}\exp\left(-\frac{|x-\tau|}{\delta\tau}\right)}{\frac{1}{2\delta\tau_e}\exp\left(-\frac{|x-\tau_e|}{\delta\tau_e}\right)} = \frac{\tau_e}{\tau}\exp\left(\frac{|x-\tau_e|}{\delta\tau_e} - \frac{|x-\tau|}{\delta\tau}\right)\\ &= \left(1 - \frac{\tau-\tau_e}{\tau}\right)\exp\left(\frac{\tau|x-\tau_e| - \tau_e|x-\tau|}{\delta\tau\tau_e}\right).\end{aligned}$$

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 4.5, 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

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

$$
\begin{aligned}
&= \left(1+\frac{K}{\tau}\right)\left(1+\frac{2K}{\delta(\tau-K)}+O\left(\frac{K^2}{\delta^2(\tau-K)^2}\right)\right)\\
&\quad-\left(1-\frac{K}{\tau}\right)\left(1-\frac{2K}{\delta(\tau-K)}-O\left(\frac{K^2}{\delta^2(\tau-K)^2}\right)\right)+2\exp\left(-\frac{1}{\delta}\right)\\
&= \frac{2K}{\tau}+\frac{4K}{\delta(\tau-K)}+O\left(\frac{K^2}{\delta^2(\tau-K)^2}\right)+2\exp\left(-\frac{1}{\delta}\right)\\
&\leq \frac{6K}{\delta(\tau-K)}+2\exp\left(-\frac{1}{\delta}\right)+O\left(\frac{K^2}{\delta^2(\tau-K)^2}\right).\\
&= O\left(\frac{K}{\delta(\tau-K)}+\exp\left(-\frac{1}{\delta}\right)\right).
\end{aligned}
$$

□

Therefore, the average sensitivity of $\mathcal{A}$ is bounded as

$$
\begin{aligned}
\beta(G) &= \mathop{\mathbb{E}}_{e\in E} d_{\mathrm{EM}}\big(\mathcal{A}(G),\mathcal{A}(G-e)\big)\\
&\leq O\left(\frac{K}{\delta(\tau-K)}+\exp\left(-\frac{1}{\delta}\right)\right)\cdot\mathsf{OPT}+\mathop{\mathbb{E}}_{L}\left[(2x-2)^2q(x)\right].
\end{aligned}
$$

We now bound the approximation guarantee of $\mathcal{A}$. By Proposition 4.5,

$$
\Pr\left[L<\left(1-\delta\ln\left(\frac{\mathsf{OPT}}{2}\right)\right)\cdot\tau(G)\right]\leq\frac{1}{\mathsf{OPT}}.
$$

Therefore, with probability at least $1-1/\mathsf{OPT}$, only those vertices with degree at least $(1-\delta\ln(\mathsf{OPT}/2))\cdot\tau(G)$ are removed from G. The number of such vertices is at most $\frac{m}{(1-\delta\ln(\mathsf{OPT}/2))\cdot\tau(G)}$. Therefore, with probability at least $1-1/\mathsf{OPT}$, the size of a maximum matching in the resulting graph is at most $\frac{m}{(1-\delta\ln(\mathsf{OPT}/2))\cdot\tau(G)}$ smaller than that of G. With probability at most $1/\mathsf{OPT}$, the size of a maximum matching in the resulting instance could be smaller by an additive term of at most OPT. Hence, the expected size of a maximum matching in the new instance is at least

$$
\mathsf{OPT}-\frac{m}{(1-\delta\ln(\mathsf{OPT}(G)/2))\cdot\tau(G)}-1.
$$

The statement on approximation guarantee follows. □

4.2.3 Average sensitivity of the greedy algorithm with thresholding

In this section, we apply Theorem 4.4 to Algorithm 2 and analyze the average sensitivity of the resulting algorithm. We show the following.

Theorem 4.8. *Let $\varepsilon\in(0,1)$ be a parameter. There exists an algorithm $\mathcal{A}_\varepsilon$ with approximation ratio $1/2-\varepsilon$ and sensitivity $O\left(\frac{\varepsilon}{1-\varepsilon}\cdot\log n+\frac{m^3}{\varepsilon^3\mathsf{OPT}(G)^3}\right)$.*

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

Algorithm $\mathcal{A}_\varepsilon$: On input $G=(V,E)$,

1. Compute OPT.

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

3. Otherwise, run the algorithm obtained by applying Theorem 4.4 to Algorithm 2 with the setting $\tau := \tau(G) = \frac{m}{\varepsilon' \mathsf{OPT}}$ and $\delta := \frac{1}{2 \ln n}$, where $\varepsilon' = \varepsilon - \frac{1}{2\mathsf{OPT}}$.

Approximation guarantee: If $\mathsf{OPT} \leq \frac{1}{\varepsilon} + 1$ or $m \leq \frac{1}{2\varepsilon}$, the approximation guarantee is clear. Otherwise, since Algorithm 2 outputs a maximal matching whose size is always at least $\frac{\mathsf{OPT}}{2}$, the size of the matching output by $\mathcal{A}_\varepsilon$ is at least $\frac{\mathsf{OPT}}{2} - \frac{\frac{\varepsilon'}{2}\mathsf{OPT}}{1 - \frac{\ln(\mathsf{OPT}/2)}{2 \ln n}} - \frac{1}{2}$, which is at least

$$\frac{\mathsf{OPT}}{2} - \varepsilon \cdot \mathsf{OPT}$$

by the setting of ε' and the fact that $\frac{\ln(\mathsf{OPT}/2)}{2 \ln n} \leq \frac{1}{2}$.

Average sensitivity: If $\mathsf{OPT} \leq \frac{1}{\varepsilon} + 1$ or $m \leq \frac{1}{2\varepsilon}$, the average sensitivity of $\mathcal{A}_\varepsilon$ is bounded by $O(\frac{1}{\varepsilon})$, 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 $\mathcal{A}_\varepsilon$ for the case that $\mathsf{OPT} > \frac{1}{\varepsilon} + 1$. Note that

$$\begin{aligned}
K_G &= 2 \max_{e \in E} \left| \frac{m}{2\varepsilon \mathsf{OPT}(G) - 1} - \frac{m-1}{2\varepsilon \mathsf{OPT}(G-e) - 1} \right| \\
&\leq 2 \max \left\{ \frac{m}{2\varepsilon \mathsf{OPT}(G) - 1} - \frac{m-1}{2\varepsilon \mathsf{OPT}(G) - 1}, \frac{m-1}{2\varepsilon(\mathsf{OPT}(G) - 1) - 1} - \frac{m}{2\varepsilon \mathsf{OPT}(G) - 1} \right\} \\
&= 2 \max \left\{ \frac{1}{2\varepsilon \mathsf{OPT}(G) - 1}, \frac{2\varepsilon(m - \mathsf{OPT}(G)) + 1}{(2\varepsilon(\mathsf{OPT}(G) - 1) - 1) \cdot (2\varepsilon \mathsf{OPT}(G) - 1)} \right\} \\
&= \frac{2}{2\varepsilon \mathsf{OPT}(G) - 1} \max \left\{ 1, \frac{2\varepsilon(m - \mathsf{OPT}(G)) + 1}{2\varepsilon(\mathsf{OPT}(G) - 1) - 1} \right\}.
\end{aligned}$$

The second inequality above uses the fact that for numbers $a, b, c \geq 0$ such that $a \leq b$ and $c(b-1) - 1 \geq 0$, we have that $\frac{a}{cb-1} \leq \frac{a-1}{c(b-1)-1}$.

From the statement of Theorem 4.3, we can see that $q(x) \leq \frac{1}{2} + x$ when $x > 0$ and $q(x) = 0$ otherwise. Therefore, we can see that the average sensitivity of the algorithm resulting from applying Theorem 4.4 to Algorithm 2 is bounded as:

$$O\left(\frac{K_G}{\delta(\tau - K_G)} + \exp\left(-\frac{1}{\delta}\right) \right) \cdot \mathsf{OPT} + \int_0^\infty (2x-2)^2 \cdot \left(\frac{1}{2} + x\right) \cdot \frac{1}{2\delta\tau} \cdot \exp\left(-\frac{|x-\tau|}{\delta\tau}\right) \mathrm{d}x. \quad (5)$$

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

$$\frac{2\varepsilon(m - \mathsf{OPT}) + 1}{2\varepsilon(\mathsf{OPT} - 1) - 1} \leq 2\varepsilon m.$$

Hence, $K_G \leq \frac{2}{2\varepsilon \mathsf{OPT} - 1} \max\{1, 2\varepsilon m\} = \frac{4\varepsilon m}{2\varepsilon \mathsf{OPT} - 1}$, since $m > \frac{1}{2\varepsilon}$ and therefore, we have that $\tau - K_G \geq \tau(1 - 2\varepsilon)$. Hence, the first term above can be upper bounded by

$$O\left(\frac{K_G}{\delta\tau(1-2\varepsilon)} + \exp\left(-\frac{1}{\delta}\right) \right) \cdot \mathsf{OPT}(G) = O\left(\frac{4\varepsilon m}{2\varepsilon \mathsf{OPT} - 1} \cdot \frac{1}{1-2\varepsilon} \cdot \frac{2 \ln n}{\frac{2m}{2\varepsilon \mathsf{OPT} - 1}} + \frac{\mathsf{OPT}}{n^2} \right)$$

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

The second term of (5) can be upper bounded by

$$\begin{aligned}
&\int_0^\infty (2x)^2 \cdot \left(\frac{1}{2} + x\right) \cdot \frac{1}{2\delta\tau} \cdot \exp\left(-\frac{|x-\tau|}{\delta\tau}\right) \mathrm{d}x \\
&= \int_0^\infty \frac{x^2}{\delta\tau} \cdot \exp\left(-\frac{|x-\tau|}{\delta\tau}\right) \mathrm{d}x + \int_0^\infty \frac{2x^3}{\delta\tau} \cdot \exp\left(-\frac{|x-\tau|}{\delta\tau}\right) \mathrm{d}x.
\end{aligned}$$

Let I_1 and I_2 denote the first and second terms above. The term I_1 can be evaluated as:

$$\begin{aligned}
I_1 &= \int_0^\tau \frac{x^2}{\delta\tau} \cdot \exp\left(-\frac{\tau - x}{\delta\tau}\right) \mathrm{d}x + \int_\tau^\infty \frac{x^2}{\delta\tau} \cdot \exp\left(-\frac{x-\tau}{\delta\tau}\right) \mathrm{d}x \\
&= \left(1 - 2\delta + 2\delta^2 - 2\exp\left(-\frac{1}{\delta}\right)\delta^2\right)\tau^2 + (1 + 2\delta + 2\delta^2)\tau^2 \\
&= \left(2 + 4\delta^2 - 2\exp\left(-\frac{1}{\delta}\right)\delta^2\right)\tau^2 = O\left(\frac{m^2}{\varepsilon^2 \mathsf{OPT}^2}\right).
\end{aligned}$$

The term I_2 can be evaluated as:

$$\begin{aligned}
I_2 &= \int_0^\tau \frac{2x^3}{\delta\tau} \cdot \exp\left(-\frac{\tau - x}{\delta\tau}\right) \mathrm{d}x + \int_\tau^\infty \frac{2x^3}{\delta\tau} \cdot \exp\left(-\frac{x-\tau}{\delta\tau}\right) \mathrm{d}x \\
&= 2\left(1 - 3\delta + 6\delta^2 - 6\delta^3 + 6\exp\left(-\frac{1}{\delta}\right)\delta^3\right)\tau^3 + 2(1 + 3\delta + 6\delta^2 + 6\delta^3)\tau^3 \\
&= 2\left(2 + 12\delta^2 + 6\exp\left(-\frac{1}{\delta}\right)\delta^3\right)\tau^3 = O\left(\frac{m^3}{\varepsilon^3 \mathsf{OPT}^3}\right).
\end{aligned}$$

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

$$\begin{aligned}
\beta(G) &= \max\left\{O\left(\frac{1}{\varepsilon}\right), O\left(\frac{\varepsilon}{1-2\varepsilon} \cdot \log n\right) + O\left(\frac{m^2}{\varepsilon^2 \mathsf{OPT}^2}\right) + O\left(\frac{m^3}{\varepsilon^3 \mathsf{OPT}^3}\right)\right\} \\
&= O\left(\frac{\varepsilon \log n}{1-\varepsilon} + \frac{m^3}{\varepsilon^3 \mathsf{OPT}^3}\right).
\end{aligned}$$

□

4.2.4 Average sensitivity of a combined matching algorithm

In this section, we combine the algorithms guaranteed by Theorems 4.1 and 4.8 in order to get a matching algorithm with improved sensitivity.

Theorem 4.9. *Let $\varepsilon \in (0, \frac{1}{2})$ be a parameter. There exists an algorithm for the maximum matching problem with approximation ratio $1/2 - \varepsilon$ and sensitivity*

$$O\left(\mathsf{OPT}^{3/4}\left(\varepsilon^{1/4}\log^{1/4} n + \frac{1}{\varepsilon^{3/4}}\right)\right).$$

In particular when $\varepsilon = 1/\log^{1/4} n$, the average sensitivity is $O(\mathsf{OPT}^{3/4}\log^{3/16} n)$.

Algorithm 3: Combined Algorithm to $\left(\frac{1}{2}-\varepsilon\right)$-Approximate Maximum Matching

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

Compute OPT.;

if $\mathsf{OPT} < 5$ *or* $m < 6$ **then**

  return an arbitrary maximum matching in G.

else

  Let $f(G) \leftarrow \frac{\mathsf{OPT}^2}{m}$ and $g(G) \leftarrow \frac{\varepsilon}{(1-\varepsilon)} \cdot \log n + \frac{m^3}{\varepsilon^3 \mathsf{OPT}^3}$;

  Run the algorithm given by Theorem 4.1 with probability $\frac{g(G)}{f(G)+g(G)}$ and run the algorithm given by Theorem 4.8 with the remaining probability.

Proof. The algorithm guaranteed by the theorem is given as Algorithm 3. The bounds on approximation guarantee and average sensitivity are both straightforward when $\mathsf{OPT} < 5$ or $m < 4$.

The approximation guarantee in the case when $\mathsf{OPT} \geq 5$ and $m \geq 6$ is also straightforward since Algorithm 3 is simply a distribution over algorithms guaranteed by Theorem 4.1 and Theorem 4.8.

We now bound the average sensitivity of Algorithm 3 when $\mathsf{OPT} \geq 5$ and $m \geq 6$. Let $\rho(G)$ denote the probability $\frac{g(G)}{f(G)+g(G)}$. By Theorem 7.2, the average sensitivity is at most

$$\frac{O(f(G)) \cdot g(G) + O(g(G)) \cdot f(G)}{f(G) + g(G)} + 2\mathsf{OPT} \cdot \mathop{\mathbb{E}}_{e \in E} \left[|\rho(G) - \rho(G-e)| \right]. \tag{6}$$

We first bound the quantity $\mathbb{E}_{e \in E} \left[|\rho(G) - \rho(G-e)| \right]$.

Claim 4.10. *For every graph $G = (V, E)$ such that $\mathsf{OPT} \geq 5$, we have, for every $e \in E$,*

$$g(G) \cdot \left(1 - \frac{3}{m}\right) \leq g(G-e) \leq g(G) \cdot \left(1 + \frac{4}{\mathsf{OPT}-1}\right).$$

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

$$\begin{aligned}
\frac{g(G-e)}{g(G)} &\leq \left(1 + \frac{\left(\frac{m-1}{\varepsilon(\mathsf{OPT}-1)}\right)^3 - \left(\frac{m}{\varepsilon \mathsf{OPT}}\right)^3}{\frac{\varepsilon \log n}{(1-\varepsilon)} + \frac{m^3}{\varepsilon^3 \mathsf{OPT}^3}}\right) \\
&\leq \left(1 + \frac{\left(\frac{m-1}{\varepsilon(\mathsf{OPT}-1)}\right)^3 - \left(\frac{m}{\varepsilon \mathsf{OPT}}\right)^3}{\frac{m^3}{\varepsilon^3 \mathsf{OPT}^3}}\right) \\
&= \left(1 - \frac{1}{m}\right)^3 \cdot \left(1 + \frac{1}{\mathsf{OPT}-1}\right)^3 \\
&\leq \left(1 + \frac{4}{\mathsf{OPT}-1}\right).
\end{aligned}$$

Note that the second-to-last inequality holds whenever $\mathsf{OPT} \geq 5$ and $m \geq 3$.
For the lower bound,

$$\frac{g(G-e)}{g(G)} \geq \left(1 - \frac{\left(\frac{m}{\varepsilon \mathsf{OPT}}\right)^3 - \left(\frac{m-1}{\varepsilon \mathsf{OPT}}\right)^3}{\frac{\varepsilon \log n}{(1-\varepsilon)} + \frac{m^3}{\varepsilon^3 \mathsf{OPT}^3}}\right)$$

$$\geq \left(1 - \frac{\left(\frac{m}{\varepsilon \mathsf{OPT}}\right)^3 - \left(\frac{m-1}{\varepsilon \mathsf{OPT}}\right)^3}{\frac{m^3}{\varepsilon^3 \mathsf{OPT}^3}}\right)$$
$$= \left(1 - \frac{1}{m}\right)^3 \geq 1 - \frac{3}{m}. \qquad \square$$

Claim 4.11. *For every graphs $G = (V, E)$ and every $e \in E$,*

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

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,

$$\frac{f(G-e)}{f(G)} \geq \left(\frac{\mathsf{OPT} - 1}{\mathsf{OPT}}\right)^2 \cdot \left(\frac{m}{m-1}\right)^2 \geq \left(\frac{\mathsf{OPT} - 1}{\mathsf{OPT}}\right)^2 \geq 1 - \frac{2}{\mathsf{OPT}}. \qquad \square$$

Note that $\left(1 - \frac{2}{\mathsf{OPT}}\right)^{-1} \leq 1 + \frac{4}{\mathsf{OPT}}$ and $\left(1 - \frac{3}{m}\right)^{-1} \leq 1 + \frac{6}{m}$ for $\mathsf{OPT} \geq 4$ and $m \geq 6$. We also have $\left(1 + \frac{4}{\mathsf{OPT}-1}\right)^{-1} \geq 1 - \frac{4}{\mathsf{OPT}-1}$ and $\left(1 + \frac{1}{m-1}\right)^{-1} \geq 1 - \frac{1}{m-1}$ for $\mathsf{OPT} \geq 5$ and $m \geq 2$.
Combining all of the above,

$$\begin{aligned}
\rho(G-e) &= \frac{g(G-e)}{f(G-e) + g(G-e)} \\
&\leq \frac{g(G) \cdot \left(1 + \frac{4}{\mathsf{OPT}-1}\right)}{(f(G) + g(G)) \cdot \min\left\{1 - \frac{3}{m}, 1 - \frac{2}{\mathsf{OPT}}\right\}} \\
&\leq \rho(G) \cdot \left(1 + \frac{4}{\mathsf{OPT} - 1}\right) \cdot \max\left\{1 + \frac{6}{m}, 1 + \frac{4}{\mathsf{OPT}}\right\} \\
&\leq \rho(G) \cdot \left(1 + \frac{12}{\mathsf{OPT} - 1}\right).
\end{aligned}$$

Using similar calculations, we can see that

$$\rho(G-e) \geq \rho(G) \cdot \left(1 - \frac{3}{m}\right) \cdot \min\left\{1 - \frac{4}{\mathsf{OPT} - 1}, 1 - \frac{1}{m-1}\right\} \geq \rho(G) \cdot \left(1 - \frac{7}{\mathsf{OPT} - 1}\right).$$

Thus, for all $e \in E$, we have that $|\rho(G) - \rho(G-e)| \leq \max\left\{\frac{7}{\mathsf{OPT}-1}, \frac{12}{\mathsf{OPT}-1}\right\} \cdot \rho(G) = \frac{12\rho(G)}{\mathsf{OPT}-1}$. Hence, $\mathbb{E}_{e \in E}[|\rho(G) - \rho(G-e)|] \leq \frac{12\rho(G)}{\mathsf{OPT}-1}$.
Therefore, the average sensitivity of Algorithm 3 is at most

$$\frac{O(f(G)) \cdot g(G) + O(g(G)) \cdot f(G)}{f(G) + g(G)} + 2\mathsf{OPT} \cdot \mathop{\mathbb{E}}_{e \in E}[|\rho(G) - \rho(G-e)|]$$

$$
\begin{aligned}
&= O\left(\frac{f(G)^{3/4}g(G)^{1/4}}{\frac{g(G)^{1/4}}{f(G)^{1/4}}+\frac{f(G)^{3/4}}{g(G)^{3/4}}}\right) + O\left(\frac{\mathsf{OPT}\rho(G)}{\mathsf{OPT}}\right)\\
&= O\left(f(G)^{3/4}g(G)^{1/4}\right) + O(1)\\
&= O\left(\left(\frac{\mathsf{OPT}^2}{m}\right)^{3/4}\cdot\left((\varepsilon\log n)^{1/4}+\left(\frac{m^3}{\varepsilon^3\mathsf{OPT}^3}\right)^{1/4}\right)\right)\\
&= O\left(\left(\frac{\mathsf{OPT}^{3/2}}{m^{3/4}}\varepsilon^{1/4}\log^{1/4}n+\frac{\mathsf{OPT}^{3/2}}{m^{3/4}}\frac{m^{3/4}}{\varepsilon^{3/4}\mathsf{OPT}^{3/4}}\right)\right)\\
&= O\left(\frac{\mathsf{OPT}^{3/2}}{m^{3/4}}\varepsilon^{1/4}\log^{1/4}n+\frac{\mathsf{OPT}^{3/4}}{\varepsilon^{3/4}}\right)\\
&= O\left(\mathsf{OPT}^{3/4}\left(\varepsilon^{1/4}\log^{1/4}n+\frac{1}{\varepsilon^{3/4}}\right)\right).
\end{aligned}
$$

□

4.3 Matching algorithm based on augmenting paths

In this section, we describe a $(1-\varepsilon)$-approximation algorithm for the maximum matching problem with average sensitivity $\tilde{O}\left(\mathsf{OPT}^{\frac{c}{c+1}}/\varepsilon^{\frac{3c}{c+1}}\right)$ for $c = O(1/\varepsilon^2)$ in Theorem 4.14. The basic building block is a $(1-\varepsilon)$-approximation algorithm for maximum matching that is based on iteratively augmenting a matching with greedily chosen augmenting paths of increasing lengths. In Theorem 4.12, we show that the average sensitivity of this algorithm is $\Delta^{O(1/\varepsilon^2)}$, where Δ is the maximum degree of the input graph. To this, we first apply Theorem 4.4 to obtain Theorem 4.13. We then combine the algorithm guaranteed by Theorem 4.13 with the algorithm guaranteed by Theorem 4.1 to obtain Theorem 4.14.

Algorithm 4: Greedy Augmenting Paths Algorithm

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

$M_0 \leftarrow \emptyset$;
for $i \in \{1, 2, \ldots \lceil \frac{1}{\varepsilon} - 1 \rceil\}$ **do**
  Let A_i denote the set of augmenting paths of length $2i-1$ for the matching M_{i-1};
  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;
  $M_i \leftarrow M_{i-1} \triangle A'_i$.
return $M_{\lceil \frac{1}{\varepsilon}-1 \rceil}$.

Theorem 4.12. *Algorithm 4 with parameter $\varepsilon > 0$ has approximation ratio $1-\varepsilon$ and average sensitivity $\Delta^{O(1/\varepsilon^2)}$, where Δ is the maximum degree of the input graph.*

Proof. For all $k \geq 0$, it is known that $|M_k| \geq \frac{k}{k+1} \cdot |M^*|$ [GJ79], where M^* denotes a maximum matching in G. Hence, the matching $M_{\lceil \frac{1}{\varepsilon}-1 \rceil}$ is a $(1-\varepsilon)$-approximation to M^*.

Yoshida et al. [YYI12, 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 Δ is the maximum degree of G. Applying Theorem 1.9 to this, we can see that the average sensitivity of Algorithm 4 with parameter $\varepsilon > 0$ and input G is $\Delta^{O(1/\varepsilon^2)}$, where Δ is the maximum degree of G. □

Theorem 4.13. *Let $\varepsilon \in (0,1)$ be a parameter. There exists an algorithm with approximation ratio $1-\varepsilon$ and average sensitivity*

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

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

Algorithm $\mathcal{A}_\varepsilon$: On input $G = (V, E)$,

1. Compute OPT.
2. If $\mathsf{OPT} \leq \frac{2}{\varepsilon} + 1$ or $m \leq \frac{1}{3\varepsilon}$, then output an arbitrary maximum matching.
3. Otherwise, run the algorithm obtained by applying Theorem 4.4 with the setting $\tau := \tau(G) = \frac{m}{\varepsilon' \mathsf{OPT}}$ and $\delta := \frac{1}{2\ln n}$ to Algorithm 4 run with parameter ε', where $\varepsilon' = \frac{\varepsilon}{3} - \frac{1}{3\mathsf{OPT}}$.

As the analysis is almost identical to that of Theorem 4.8, we only highlight the differences.

Approximation guarantee: The analysis of the approximation ratio is straightforward.

Average sensitivity: We focus on the case that $\mathsf{OPT} > \frac{2}{\varepsilon}+1$ and $m > \frac{1}{3\varepsilon}$ since the average sensitivity in other case is straightforward to analyze. The first term of (5) can be bounded by $O\left(\frac{\varepsilon}{1-\varepsilon} \cdot \log n\right)$ in the same way as in the proof of Theorem 4.8.

Let $c = O(1/\varepsilon^2)$. Then, the second term of (5) becomes

$$\int_0^\infty (2x-2)^2 x^c \frac{1}{2\delta\tau} \exp\left(-\frac{|x-\tau|}{\delta\tau}\right) \mathrm{d}x = 4\int_\tau^\infty x^{c+2} \frac{1}{\delta\tau} \exp\left(-\frac{x-\tau}{\delta\tau}\right) \mathrm{d}x$$
$$= \exp\left(\frac{1}{\delta}\right)(\delta\tau)^{c+2}\Gamma\left(c+3, \frac{1}{\delta}\right) = (\delta\tau)^{c+2}(c+2)! \sum_{k=0}^{c+2} \frac{(1/\delta)^k}{k!} = \left(\frac{m}{\varepsilon^3 \mathsf{OPT}}\right)^{O(1/\varepsilon^2)}$$

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

By combining Theorems 4.1 and 4.13, we get the following.

Theorem 4.14. *Let $\varepsilon \in (0,1)$ be a parameter. There exists an algorithm with approximation ratio $1-\varepsilon$ and average sensitivity*

$$\mathrm{OPT}(G)^{\frac{c}{c+1}} \cdot O\left(\left(\frac{\varepsilon}{1-\varepsilon} \cdot \log n\right)^{\frac{1}{c+1}} + \frac{1}{\varepsilon^{\frac{3c}{c+1}}}\right)$$

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

Proof. Let $f(G) = \frac{\mathsf{OPT}^2}{m}$ and $g(G) = \frac{\varepsilon}{1-\varepsilon} \cdot \log n + \left(\frac{m}{\varepsilon^3 \mathsf{OPT}}\right)^c$ for $c = O(1/\varepsilon^2)$. Let $\rho(G)$ denote $\frac{g(G)}{f(G)+g(G)}$. Given a graph $G = (V, E)$ as input and a parameter $\varepsilon \in (0, 1)$, the algorithm guaranteed by the theorem first computes OPT and returns an arbitrary maximum matching if $\mathsf{OPT} < 2c$ or $m < 2c$. Otherwise, it runs the algorithm given by Theorem 4.1 with probability $\rho(G)$ and the algorithm given by Theorem 4.13 using the parameter ε with probability $1 - \rho(G)$.

We highlight the differences in the analysis when compared to the proof of Theorem 4.9, which arise only in the part where we analyze the average sensitivity for the case that $\mathsf{OPT} \geq 2c$ and $m \geq 2c$. While bounding the second term of (6), the bounds on $f(G - e)$ that we use are identical to that in Claim 4.11. The following claim gives bounds for $g(G - e)$.

Claim 4.15. *For every graph $G = (V, E)$ such that $\mathsf{OPT} \geq c+1$ and $m \geq 2c$, and for every $e \in E$,*

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

The proof of Claim 4.15 is nearly identical to that of Claim 4.10. In order to get the upper bound, we use the fact that $(1 + x)^r \leq 1 + \frac{rx}{1-(r-1)x}$ for $x \in [0, \frac{1}{r-1})$ and $r > 1$.

Using these bounds, we can argue that $\mathbb{E}_{e \in E}[|\rho(G) - \rho(G - e)|] \leq \frac{3c}{\mathsf{OPT}-c}$. Therefore, the second term of (6) can be bounded by $\frac{3c\mathsf{OPT}}{\mathsf{OPT}-c}$. This is $O(1/\varepsilon^2)$, since $\frac{\mathsf{OPT}}{\mathsf{OPT}-c} \leq 2$ as $\mathsf{OPT} \geq 2c$.

To bound the first term of (6), we divide both the numerator and denominator by $f(G)^{\frac{1}{c+1}} \cdot g(G)^{\frac{c}{c+1}}$ and upper bound the resulting fraction by its numerator $g(G)^{\frac{1}{c+1}} \cdot f(G)^{\frac{c}{c+1}}$, which can be simplified to obtain the final upper bound on the average sensitivity. □

5 Minimum Vertex Cover

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. In this section, we discuss averagely stable approximation algorithms for the minimum vertex cover problem.

In Section 5.1, we give an algorithm that reduces to the maximum matching problem and prove Theorem 5.1. Then, in Section 5.2, we show another algorithm based on a differentially private algorithm due to Gupta et al. [GLM+10], which has a worse approximation ratio but could have a smaller average sensitivity compared to the first algorithm. In particular, we prove Theorem 5.2.

5.1 Reduction to the maximum matching problem

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. Based on this fact, we slightly modify Algorithm 3 to obtain an averagely stable algorithm for the minimum vertex cover problem. Specifically, we show the following.

Theorem 5.1. *Let $\varepsilon \in (0, 1)$. There exists a $(2 + \varepsilon)$-approximation algorithm for the minimum vertex cover problem with average sensitivity*

$$O\left(\mathsf{OPT}^{3/4}\left(\varepsilon^{1/4} \log^{1/4} n + \frac{1}{\varepsilon^{3/4}}\right)\right).$$

In particular when $\varepsilon = 1/\log^{1/4} n$, the average sensitivity is $O(\mathsf{OPT}^{3/4} \log^{3/16} n)$.

Proof. Given a graph $G = (V, E)$, let MM denote the size of a maximum matching in G. Let $\mathcal{A}$ denote the algorithm given by Theorem 4.9. Our algorithm to approximate vertex cover is a slight modification of $\mathcal{A}$.

Recall that, on input $G = (V, E)$ and parameter $\varepsilon \in (0, 1/2)$, algorithm $\mathcal{A}$ runs one of the following algorithms. The first one, denoted by $\mathcal{A}_1$, simply outputs a maximum matching in G. The second one, denoted by $\mathcal{A}_2$, does the following. If $\mathsf{MM} \leq \frac{1}{\varepsilon} + 1$ or $m \leq \frac{1}{2\varepsilon}$, it outputs an arbitrary maximum matching. Otherwise, $\mathcal{A}_2$ constructs a graph $[G]_L$ by removing vertices of degrees at least a threshold $L \sim \mathsf{Lap}(\tau, \delta\tau)$ and then applies the randomized greedy algorithm on $[G]_L$, where $\tau = \frac{2m}{2\varepsilon\mathsf{MM}-1}$, and $\delta = \frac{1}{2\ln n}$.

Our modification to $\mathcal{A}$ is as follows. When we run $\mathcal{A}_1$, we output the vertex set S consisting of the endpoints of the output matching. When we run $\mathcal{A}_2$, we output the set $T = T_1 \cup T_2$, where T_1 is the set of endpoints of the matching and T_2 is the set of vertices removed by $\mathcal{A}_2$.

Approximation guarantee: Clearly, S is a 2-approximate vertex cover. As T_1 is a vertex cover of $[G]_L$, the set T is a vertex cover of G. We now bound the expected size of T. Let OPT and $\mathsf{OPT}([G]_L)$ be the sizes of minimum vertex covers in G and $[G]_L$, respectively. Then, observing that $\mathsf{OPT} \geq \mathsf{OPT}([G]_L) \geq |T_1|/2$, we have $\mathbb{E}[|T_1|] \leq 2\mathsf{OPT}$. We now bound $\mathbb{E}[|T_2|]$. By Proposition 4.5, we know that the value of $L < (1 - \frac{\ln 2}{2}) \cdot \tau$ with probability at most $\frac{1}{n}$. Therefore, with probability at least $1 - \frac{1}{n}$, the number of vertices removed is at most $\frac{m}{(1-\frac{\ln 2}{2})\cdot\tau}$ and with probability at most $\frac{1}{n}$, the number of vertices removed is at most n. Therefore, the expected cardinality of S_2 is at most $\frac{m}{(2-\ln 2)\cdot\frac{m}{2\varepsilon\mathsf{MM}-1}} \cdot (1 - \frac{1}{n}) + n \cdot \frac{1}{n}$, which is at most $2\varepsilon\mathsf{MM}$. Note that $\frac{\mathsf{OPT}}{2} \leq \mathsf{MM} \leq \mathsf{OPT}$ as the set of endpoints in a maximum matching is a 2-approximate vertex cover. Therefore, we can see that $\mathbb{E}[|T_2|] \leq 2\varepsilon\mathsf{OPT}$. It follows that

$$\mathbb{E}[|T|] = \mathbb{E}[|T_1|] + \mathbb{E}[|T_2|] \leq 2\mathsf{OPT} + 2\varepsilon\mathsf{OPT} = (2 + 2\varepsilon) \cdot \mathsf{OPT}.$$

Average sensitivity: The average sensitivity of $\mathcal{A}_1$ even after the modification is $O\left(\frac{\mathsf{OPT}^2}{m}\right)$, since outputting the endpoints of a matching instead of the matching itself can only affect the average sensitivity by a factor of at most 2.

We now bound the average sensitivity of $\mathcal{A}_2$ after the modification. Consider a graph $G = (V, E)$. Let A_2^i for $i \in \{1, 2\}$ denote the algorithm that simulates the actions of $\mathcal{A}_2$ and outputs only T_i. Let $H_{e,\pi}$ for $e \in E$ denote the Hamming distance between the outputs of $\mathcal{A}_2$ on G and $G - e$ when run using the same random string $\pi \in \{0, 1\}^*$. Let $H^i_{e,\pi}$ for $i \in \{1, 2\}$ and $e \in E$ denote the Hamming distance between the outputs of $\mathcal{A}_2^{(i)}$ on G and $G - e$ when run using the same random string π. Since T_1 and T_2 are always disjoint, we have for all $e \in E$ and all $\pi \in \{0, 1\}^*$,

$$H_{e,\pi} \leq H^1_{e,\pi} + H^2_{e,\pi}.$$

Therefore, the average sensitivity of $\mathcal{A}_2$ is at most the sum of average sensitivities of $\mathcal{A}_2^{(1)}$ and $\mathcal{A}_2^{(2)}$.

The average sensitivity of $\mathcal{A}_2^{(1)}$ is $O\left(\frac{\varepsilon}{1-\varepsilon} \cdot \log n + \frac{m^3}{\varepsilon^3\mathsf{MM}^3}\right)$, since outputting the endpoints of a matching does not change the average sensitivity asymptotically.

We now bound the average sensitivity of $\mathcal{A}_2^{(2)}$. Note that the output of $\mathcal{A}_2^{(2)}$ on a graph $G = (V, E)$ is fully characterized by the value of the Laplace random variable that is sampled. Fix $e \in E$. We bound the earth mover's distance between $\mathcal{A}_2^{(2)}(G)$ and $\mathcal{A}_2^{(2)}(G - e)$.

Let $f_G(x)$ and $f_{G-e}(x)$ denote the probability densities that $\mathcal{A}_2^{(2)}$ samples the value x as the threshold on inputs G and $G - e$, respectively. We start with the distribution $\mathcal{A}_2^{(2)}(G)$. For

each $x \in \mathbb{R}$, we retain a probability density of $\min\{f_{G-e}(x), f_G(x)\}$ at point x at a cost of $2 \cdot \min\{f_{G-e}(x), f_G(x)\}$. The factor 2 comes from the fact that for the same value of random threshold, the sets output by $\mathcal{A}_2^{(2)}$ on G and $G-e$ can differ in at most 2 vertices.

We also transport a probability density of $\max\{f_G(x) - f_{G-e}(x), 0\}$ to another point where there is a deficit in probability density. The cost of this transport equals to the transported probability density weighted by Hamming distance between the cardinality of outputs at the source and destination points. Let MM_e denote the size of a maximum matching in $G-e$. As we argued earlier, for the distribution $\mathcal{A}_2^{(2)}(G)$, a probability of at least $1 - \frac{1}{n}$ is located on values of x that would make the cardinality of output of $\mathcal{A}_2^{(2)}$ at most $2\varepsilon\mathsf{MM}$. Using the same argument, we can see that, for the distribution $\mathcal{A}_2^{(2)}(G-e)$, a probability of at least $1 - \frac{1}{n}$ is located on values of x that would make the cardinality of output of $\mathcal{A}_2^{(2)}$ at most $2\varepsilon\mathsf{MM}_e \leq 2\varepsilon\mathsf{MM}$. If a probability p is transported from a source point with output size at most $2\varepsilon\mathsf{MM}$ to a destination point with output size at most $2\varepsilon\mathsf{MM}$, the cost of the transport is at most $p \cdot 4\varepsilon\mathsf{MM}$. Only for a probability of at most $\frac{2}{n}$, the symmetric difference between the source and destination output sets is $\Omega(n)$ and hence, the cost of transport is at most 2. Thus, the earth mover's distance between $\mathcal{A}_2^{(2)}(G)$ and $\mathcal{A}_2^{(2)}(G-e)$ is at most $d_{\mathrm{TV}}(L, L_e) \cdot 4\varepsilon\mathsf{MM} + O(1)$.

Using Claim 4.7 and following the steps in the proof of Theorem 4.8, the average sensitivity of $\mathcal{A}_2^{(2)}$ is bounded by $O(\frac{\varepsilon^2}{1-\varepsilon}\log n)$.

Hence, the average sensitivity of $\mathcal{A}_2$ is $O\left(\frac{\varepsilon}{1-\varepsilon} \cdot \log n + \frac{m^3}{\varepsilon^3 \mathsf{MM}^3}\right)$. Following the proof of Theorem 4.9 again, the overall average sensitivity is bounded by

$$O\left(\mathsf{MM}^{3/4}\left(\varepsilon^{1/4}\log^{1/4} n + \frac{1}{\varepsilon^{3/4}}\right)\right) = O\left(\mathsf{OPT}^{3/4}\left(\varepsilon^{1/4}\log^{1/4} n + \frac{1}{\varepsilon^{3/4}}\right)\right).$$

By replacing ε with $\varepsilon/2$ throughout, we get the approximation and average sensitivity guarantees given by the statement. □

5.2 Algorithm based on a differentially private algorithm

We consider another algorithm for the minimum vertex cover problem based on the differentially private algorithm due to Gupta et al. [GLM+10] and show the following.

Theorem 5.2. *Let $\varepsilon \geq 0$. There exists an algorithm for the minimum vertex cover problem with average sensitivity $O(n^2/m^{1+\varepsilon})$ and approximation ratio $O((m^{1+\varepsilon}\log n)/n)$.*

The algorithm of Gupta et al. [GLM+10] is based on the simple (non-private) 2-approximation algorithm [Pit85] that repeatedly selects a vertex at random with probability proportional to its degree with respect to the uncovered edges and adds the sampled vertex to the solution. To make this algorithm differentially private, Gupta et al. [GLM+10] mixed the distribution with a uniform distribution, using a weight that grows as the number of remaining vertices decreases. Our algorithm, shown in Algorithm 5, is similar to theirs though our choice of weights is different.

We will show that Algorithm 5 has approximation ratio $O((m^{1+\varepsilon}\log n)/n)$ and average sensitivity $O(n^2/m^{1+\varepsilon})$. Although the approximation ratio is worse than that discussed in Section 5.1, the present one has a better sensitivity. In what follows, for simplicity, we assume $n \leq m^{1+\varepsilon}$.

Lemma 5.3. *The approximation ratio of Algorithm 5 is $O((m^{1+\varepsilon}\log n)/n)$.*

Algorithm 5: AVERAGELY STABLE ALGORITHM FOR VERTEX COVER

Input: Graph $G = (V, E)$ and parameter $\varepsilon \geq 0$

Let $G_1 = (V_1, E_1) \leftarrow G$, and $S \leftarrow \emptyset$;
for $i = 1, \ldots, n$ **do**
  $w_i \leftarrow 2m^{1+\varepsilon}/(n - i + 1)$;
  Sample a vertex $v \in V(G_i)$ with probability proportional to $d_{G_i}(v) + w_i$;
  if $d_{G_i}(v) \geq 1$ **then**
    $S \leftarrow S \cup \{v\}$.
  $G_{i+1} \leftarrow$ the graph obtained from G_i by removing v and incident edges.
return S.

Proof. It is shown in [GLM+10] that the approximation ratio is $(2 + (2/n)\sum_{i=1}^{n} w_i)$, which is

$$2 + \frac{2}{n}\sum_{i=1}^{n} \frac{2m^{1+\varepsilon}}{n - i + 1} = O\left(\frac{m^{1+\varepsilon}\log n}{n}\right). \qquad \square$$

Lemma 5.4. *The average sensitivity of Algorithm 5 is $O(n^2/m^{1+\varepsilon})$.*

Proof. For a while, we fix a graph $G = (V, E)$ and an edge $e \in E$. To analyze the sensitivity of Algorithm 5, we focus on the vertex ordering $\pi = (v_1, \ldots, v_n)$, where v_i $(i = 1, \ldots, n)$ is the vertex sampled at the i-th step. Note that the output set is fully determined by π. Let $p_G(\pi)$ and $p_{G-e}(\pi)$ denote the probabilities that Algorithm 5 samples the vertex ordering π on inputs G and $G - e$, respectively. Then, we can bound the earth mover's distance between $\mathcal{A}(G)$ and $\mathcal{A}(G - e)$, where $\mathcal{A}$ is Algorithm 5, as follows. For each vertex ordering π, we retain a mass of $p_G(\pi)$ at a cost of $1 \cdot p_G(\pi)$ (the output sets can have a Hamming distance of 1 even if the vertex orderings generated are the same) and bring in a "remaining" mass of $\max\{0, p_{G-e}(\pi) - p_G(\pi)\}$ from other points where there is excess probability mass. The second step costs at most $n \cdot \max\{0, p_{G-e}(\pi) - p_G(\pi)\}$.

We now bound $p_{G-e}(\pi) - p_G(\pi)$. Let $k_e \in \{1, \ldots, n\}$ be such that the first endpoint of e in π occurs in the k_e-th position in π. As long as e is fixed, we write k instead for simplicity. Let G_i and G'_i be the graphs at the beginning of the i-th step of Algorithm 5 on G and $G - e$, respectively. Let m_i and m'_i be the number of edges in G_i and G'_i, respectively. Note that $m_i = m'_i + 1$ for $i \leq k$ and $m_i = m'_i$ for $i > k$. For a vertex $v \in V$, let $d_i(v)$ and $d'_i(v)$ denote the degrees of v in G_i and G'_i, respectively. Note that $d_i(v_i) = d'_i(v_i)$ for $i \neq k$ and $d_k(v_k) = d'_k(v_k) + 1$. Now, we have

$$\begin{aligned}
&p_{G-e}(\pi) - p_G(\pi) \\
&= \prod_{i=1}^{n} \frac{d'_i(v_i) + w_i}{(n - i + 1)w_i + 2m'_i} - \prod_{i=1}^{n} \frac{d_i(v_i) + w_i}{(n - i + 1)w_i + 2m_i} \\
&= \frac{\prod_{i \in [n]: i \neq k} d_i(v_i) + w_i}{\prod_{i=k+1}^{n}(n - i + 1)w_i + 2m_i} \left[\frac{d'_k(v_k) + w_k}{\prod_{i=1}^{k}(n - i + 1)w_i + 2m'_i} - \frac{d_k(v_k) + w_k}{\prod_{i=1}^{k}(n - i + 1)w_i + 2m_i}\right] \\
&= \frac{\prod_{i \in [n]: i \neq k} d_i(v_i) + w_i}{\prod_{i=k+1}^{n}(n - i + 1)w_i + 2m_i} \left[\frac{d_k(v_k) - 1 + w_k}{\prod_{i=1}^{k}(n - i + 1)w_i + 2m_i - 2} - \frac{d_k(v_k) + w_k}{\prod_{i=1}^{k}(n - i + 1)w_i + 2m_i}\right].
\end{aligned}$$

Let Z_π and Z'_π stand for $\prod_{i\le k}(n-i+1)w_i+2m_i$ and $\prod_{i\le k}(n-i+1)w_i+2m_i-2$, respectively. The expression inside the square brackets can be written as

$$(d_k(v_k)+w_k)\left(\frac{1}{Z'_\pi}-\frac{1}{Z_\pi}\right)-\frac{1}{Z'_\pi}.$$

Let t_i stand for $(n-i+1)w_i+2m_i$ for all $i\le k$. By using the identity

$$\prod_{i=1}^{n}x_i-\prod_{i=1}^{n}y_i=\sum_{i=1}^{n}(x_i-y_i)\cdot\prod_{j=1}^{i-1}x_j\cdot\prod_{j=i+1}^{n}y_j,$$

we can rewrite $\frac{1}{Z'_\pi}-\frac{1}{Z_\pi}$ as follows.

$$\begin{aligned}\frac{1}{Z'_\pi}-\frac{1}{Z_\pi}&=\sum_{i=1}^{k}\left(\prod_{j=1}^{i-1}\frac{1}{t_j-2}\right)\left(\frac{1}{t_i-2}-\frac{1}{t_i}\right)\left(\prod_{j=i+1}^{k}\frac{1}{t_j}\right)=2\sum_{i=1}^{k}\left(\prod_{j=1}^{i}\frac{1}{t_j-2}\right)\left(\prod_{j=i}^{k}\frac{1}{t_j}\right)\\&\le 2\prod_{i=1}^{k}\frac{1}{t_i-2}\sum_{i=1}^{k}\frac{1}{t_i}=\frac{2}{Z'_\pi}\sum_{i=1}^{k}\frac{1}{t_i}=\frac{2}{Z_\pi}\cdot\frac{Z_\pi}{Z'_\pi}\cdot\sum_{i=1}^{k}\frac{1}{t_i}.\end{aligned}$$

Note that

$$\begin{aligned}\frac{Z_\pi}{Z'_\pi}&=\prod_{i=1}^{k}\frac{(n-i+1)w_i+2m_i}{(n-i+1)w_i+2m_i-2}=\prod_{i=1}^{k}\left(1+\frac{2}{(n-i+1)w_i+2m_i-2}\right)\\&\le\exp\left(\sum_{i=1}^{k}\frac{2}{(n-i+1)w_i+2m_i-2}\right)=\exp\left(\sum_{i=1}^{k}\frac{2}{2m^{1+\varepsilon}+2m_i-2}\right)\\&\le\exp\left(\frac{n}{m^{1+\varepsilon}}\right)\le e.\qquad\text{(From the assumption } n\le m^{1+\varepsilon}.)\end{aligned}$$

Combining all of the above, we get,

$$\begin{aligned}p_{G-e}(\pi)-p_G(\pi)&=\frac{\prod_{i\in[n]:i\neq k}d_i(v_i)+w_i}{\prod_{i=k+1}^{n}(n-i+1)w_i+2m_i}\left[(d_k(v_k)+w_k)\left(\frac{1}{Z'_\pi}-\frac{1}{Z_\pi}\right)-\frac{1}{Z'_\pi}\right]\\&\le\frac{\prod_{i\in[n]:i\neq k}d_i(v_i)+w_i}{\prod_{i=k+1}^{n}(n-i+1)w_i+2m_i}\left[(d_k(v_k)+w_k)\left(\frac{2}{Z_\pi}\cdot\frac{Z_\pi}{Z'_\pi}\cdot\sum_{i\le k}\frac{1}{t_i}\right)\right]\\&\le 2ep_G(\pi)\sum_{i\le k}\frac{1}{t_i}.\end{aligned}$$

Now, we take the average over $e\in E$.

$$\begin{aligned}\mathop{\mathbb{E}}_{e}\left[p_{G-e}(\pi)-p_G(\pi)\right]&\le\frac{1}{m}\sum_{e\in E}\left(2ep_G(\pi)\sum_{i=1}^{k_e}\frac{1}{t_i}\right)=\frac{2ep_G(\pi)}{m}\sum_{j=1}^{n}\left(d_j(v_j)\sum_{i=1}^{j}\frac{1}{t_i}\right)\\&=\frac{2ep_G(\pi)}{m}\sum_{i=1}^{n}\frac{1}{t_i}\sum_{j=i}^{n}d_j(v_j)=\frac{2ep_G(\pi)}{m}\sum_{i=1}^{n}\frac{m_i}{t_i}\end{aligned}$$

$$= \frac{2ep_G(\pi)}{m} \sum_{i=1}^{n} \frac{m_i}{(n-i+1)w_i + 2m_i} \leq \frac{2ep_G(\pi)}{m} \sum_{i=1}^{n} \frac{m_i}{2m^{1+\varepsilon} + 2m_i}$$
$$\leq \frac{ep_G(\pi)}{m} \sum_{i=1}^{n} \frac{1}{m^{\varepsilon}} \leq \frac{enp_G(\pi)}{m^{1+\varepsilon}}.$$

Then, the average sensitivity is bounded by

$$1 + n \sum_{\pi} \mathop{\mathbb{E}}_{e} \left[p_{G-e}(\pi) - p_G(\pi) \right] = 1 + \sum_{\pi} \frac{en^2 p_G(\pi)}{m^{1+\varepsilon}} = O\left(\frac{n^2}{m^{1+\varepsilon}} \right). \qquad \square$$

6 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 $f : V \to \{0, 1\}$ 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 averagely stable algorithm for the 2-coloring problem.

Theorem 6.1. *Any (randomized) algorithm for the* 2*-coloring problem has average sensitivity* $\Omega(n)$*.*

Proof. Suppose that there is a (randomized) algorithm $\mathcal{A}$ whose average sensitivity is at most βn 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 $d_{\mathrm{EM}}\big(\mathcal{A}(P), \mathcal{A}(Q)\big) \leq 8\beta n$. 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

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

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.

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 $d_{\mathrm{Ham}}(f_i, f_j) \geq n/16$ for

$i \neq j$ because Q is heavy. Let $q_i = \Pr[\mathcal{A}(Q^*) = f_i]$ for $i \in [4]$. As the edge (P_1, Q^*) is intimate, we have

$$\begin{aligned} 8\beta n &\geq d_{\mathrm{EM}}\big(\mathcal{A}(P_1), \mathcal{A}(Q^*)\big) \geq \frac{n}{16}\left(\big|\Pr[\mathcal{A}(P_1) = f_1] - q_1\big| + \big|\Pr[\mathcal{A}(P_1) = f_2] - q_2\big| + q_3 + q_4\right) \\ &= \frac{n}{16}\left(\big|\Pr[\mathcal{A}(P_1) = f_1] - q_1\big| + \big|\Pr[\mathcal{A}(P_1) = f_2] - q_2\big| + 1 - q_1 - q_2\right) \end{aligned}$$

and hence we must have $q_1 + q_2 \geq 1 - 128\beta$. Considering $d_{\mathrm{EM}}\big(\mathcal{A}(P_2), \mathcal{A}(Q^*)\big)$, 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. □

7 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.

7.1 k-average sensitivity from average sensitivity

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

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

Proof. We have

$$\begin{aligned} &\mathop{\mathbb{E}}_{e_1,\dots,e_k \in E}\left[d_{\mathrm{EM}}\big(\mathcal{A}(G), \mathcal{A}(G - \{e_1, \dots, e_k\})\big)\right] \\ \leq &\mathop{\mathbb{E}}_{e_1,\dots,e_k \in E}\left[\sum_{i=1}^{k} d_{\mathrm{EM}}\big(\mathcal{A}(G - \{e_1, \dots, e_{i-1}\}), \mathcal{A}(G - \{e_1, \dots, e_i\})\big)\right] \\ = &\mathop{\mathbb{E}}_{e_1 \in E}\Big[d_{\mathrm{EM}}\big(\mathcal{A}(G), \mathcal{A}(G - \{e_1\})\big)\Big) + \mathop{\mathbb{E}}_{e_2 \in E}\Big[d_{\mathrm{EM}}\Big(\mathcal{A}(G - \{e_1\}), \mathcal{A}(G - \{e_1, e_2\})\big) + \cdots \\ &\qquad + \mathop{\mathbb{E}}_{e_k \in E}\Big[d_{\mathrm{EM}}\big(\mathcal{A}(G - \{e_1, \dots, e_{k-1}\}), \mathcal{A}(G - \{e_1, \dots, e_k\})\Big)\dots\Big]\Big]\Big] \\ = &f(n, m) + \mathop{\mathbb{E}}_{e_1 \in E}\Big[\beta(G - \{e_1\}) + \mathop{\mathbb{E}}_{e_2 \in E}\Big[\beta(G - \{e_1, e_2\}) + \cdots + \mathop{\mathbb{E}}_{e_{k-1} \in E}\Big[\beta(G - \{e_1, \dots, e_{k-1}\})\dots\Big]\Big]\Big] \\ \leq &\sum_{i=1}^{k} f(n, m - i + 1). \end{aligned}$$

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

7.2 Sequential composability

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

Theorem 1.6 (Sequential composability)**.** *Consider two randomized algorithms $\mathcal{A}_1 : \mathcal{G} \to \mathcal{S}_1, \mathcal{A}_2 : \mathcal{G} \times \mathcal{S}_1 \to \mathcal{S}_2$. Suppose that the average sensitivity of $\mathcal{A}_1$ with respect to the total variation distance is γ_1 and the average sensitivity of $\mathcal{A}_2(\cdot, S_1)$ is $\beta_2^{(S_1)}$ for any $S_1 \in \mathcal{S}_1$. Let $\mathcal{A} : \mathcal{G} \to \mathcal{S}_2$ be a randomized algorithm obtained by composing $\mathcal{A}_1$ and $\mathcal{A}_2$, that is, $\mathcal{A}(G) = \mathcal{A}_2(G, \mathcal{A}_1(G))$. Then, the average sensitivity of $\mathcal{A}$ is $\mathsf{H} \cdot \gamma_1(G) + \mathbb{E}_{S_1 \sim \mathcal{A}_1(G)}\left[\beta_2^{(S_1)}(G)\right]$, where H denotes the maximum Hamming weight among those of solutions obtained by running $\mathcal{A}$ on G and $\{G - e\}_{e \in E}$.*

Proof. Consider $G = (V, E)$ and let $e \in E$. We bound the earth mover's distance between $\mathcal{A}(G)$ and $\mathcal{A}(G - e)$ as follows. For a distribution $\mathcal{D}$, we use $f_{\mathcal{D}}$ to denote its probability mass function. We know that for all $S_1 \in \mathcal{S}_1$ and $S_2 \in \mathcal{S}_2$

$$f_{(\mathcal{A}_1(G), \mathcal{A}_2(G, S_1))}(S_1, S_2) = f_{\mathcal{A}_1(G)}(S_1) \cdot f_{\mathcal{A}_2(G, S_1)}(S_2),$$

where $(\mathcal{A}_1(G), \mathcal{A}_2(G, S_1))$ denotes the joint distribution of $\mathcal{A}_1(G)$ and $\mathcal{A}_2(G, S_1)$. Fix $S_1 \in \mathcal{S}_1$. For each $S_2 \in \mathcal{S}_2$, we transform probabilities of the form $f_{(\mathcal{A}_1(G), \mathcal{A}_2(G, S_1))}(S_1, S_2)$ to $f_{\mathcal{A}_1(G)}(S_1) \cdot f_{\mathcal{A}_2(G-e, S_1)}(S_2)$. This incurs a total cost of $f_{\mathcal{A}_1(G)}(S_1) \cdot d_{\mathrm{EM}}(\mathcal{A}_2(G, S_1), \mathcal{A}_2(G - e, S_1))$. We can now, for each $S_1 \in \mathcal{S}_1$ and $S_2 \in \mathcal{S}_2$, transform the probability $f_{\mathcal{A}_1(G)}(S_1) \cdot f_{\mathcal{A}_2(G-e, S_1)}(S_2)$ into $f_{\mathcal{A}_1(G-e)}(S_1) \cdot f_{\mathcal{A}_2(G-e, S_1)}(S_2)$ at a cost of at most $d_{\mathrm{TV}}(\mathcal{A}_1(G), \mathcal{A}_1(G - e)) \cdot \mathsf{H}$, where H denotes the maximum Hamming weight among those of solutions obtained by running $\mathcal{A}$ on G and $\{G - e\}_{e \in E}$. Thus, the earth mover's distance between $\mathcal{A}(G)$ and $\mathcal{A}(G - e)$ is at most

$$d_{\mathrm{TV}}(\mathcal{A}_1(G), \mathcal{A}_1(G - e)) \cdot \mathsf{H} + \int_{\mathcal{S}_1} f_{\mathcal{A}_1(G)}(S_1) \cdot d_{\mathrm{EM}}\big(\mathcal{A}_2(G, S_1), \mathcal{A}_2(G - e, S_1)\big)\, \mathrm{d}S_1.$$

Hence, the average sensitivity of $\mathcal{A}$ can be bounded as:

$$\begin{aligned}
\mathop{\mathbb{E}}_{e \in E}\left[d_{\mathrm{EM}}(\mathcal{A}(G), \mathcal{A}(G - e))\right] &\leq \mathsf{H} \cdot \mathop{\mathbb{E}}_{e \in E}\left[d_{\mathrm{TV}}(\mathcal{A}_1(G), \mathcal{A}_1(G - e))\right] \\
&\quad + \mathop{\mathbb{E}}_{e \in E}\left[\int_{S_1 \in \mathcal{S}_1} f_{\mathcal{A}_1(G)}(S_1) \cdot d_{\mathrm{EM}}(\mathcal{A}_2(G, S_1), \mathcal{A}_2(G - e, S_1))\, \mathrm{d}S_1\right] \\
&\leq \mathsf{H}\gamma_1(G) + \mathop{\mathbb{E}}_{S_1 \sim \mathcal{A}_1(G)}\left[d_{\mathrm{EM}}(\mathcal{A}_2(G, S_1), \mathcal{A}_2(G - e, S_1))\right] \\
&= \mathsf{H}\gamma_1(G) + \mathop{\mathbb{E}}_{S_1 \sim \mathcal{A}_1(G)}\left[\mathop{\mathbb{E}}_{e \in E} d_{\mathrm{EM}}(\mathcal{A}_2(G, S_1), \mathcal{A}_2(G - e, S_1))\right] \\
&= \mathsf{H}\gamma_1(G) + \mathop{\mathbb{E}}_{S_1 \sim \mathcal{A}_1(G)}\left[\beta_2^{(S_1)}(G)\right].
\end{aligned}$$

We are able to interchange the order of expectations because of Fubini's theorem [Fub07]. □

The following theorem states the composability of average sensitivity with respect to the total variation distance.

Theorem 1.7 (Sequential composability w.r.t. the TV distance)**.** *Consider k randomized algorithms $\mathcal{A}_i : \mathcal{G} \times \prod_{j=1}^{i-1} \mathcal{S}_j \to \mathcal{S}_i$ for $i \in \{1, \ldots, k\}$. Suppose that, for each $i \in \{1, \ldots, k\}$, the average*

sensitivity of $\mathcal{A}_i(\cdot, S_1, \ldots, S_{i-1})$ is γ_i with respect to the total variation distance for every $S_1 \in \mathcal{S}_1, \ldots, S_{i-1} \in \mathcal{S}_{i-1}$. Consider a sequence of computations $S_1 = \mathcal{A}_1(G), S_2 = \mathcal{A}_2(G, S_1), \ldots, S_k = \mathcal{A}_k(G, S_1, \ldots, S_{k-1})$. Let $\mathcal{A} : \mathcal{G} \to \mathcal{S}_k$ be a randomized algorithm that performs this sequence of computations on input G and outputs S_k. Then, the average sensitivity of $\mathcal{A}$ is at most $\sum_{i=1}^{k} \gamma_i(G)$ with respect to the total variation distance.

Theorem 1.7 can be immediately obtained by iteratively applying Lemma 7.1.

Lemma 7.1. *Consider two randomized algorithms $\mathcal{A}_1 : \mathcal{G} \to \mathcal{S}_1, \mathcal{A}_2 : \mathcal{G} \times \mathcal{S}_1 \to \mathcal{S}_2$ for a graph problem. Suppose that the average sensitivity of $\mathcal{A}_1$ is $\gamma_1(G)$ and the average sensitivity of $\mathcal{A}_2(\cdot, S_1)$ is $\gamma_2(G)$ for any $S_1 \in \mathcal{S}_1$, both with respect to the total variation distance. Let $\mathcal{A} : \mathcal{G} \to \mathcal{S}_2$ be a randomized algorithm obtained by composing $\mathcal{A}_1$ and $\mathcal{A}_2$, that is, $\mathcal{A}(G) = \mathcal{A}_2(G, \mathcal{A}_1(G))$. Then, the average sensitivity of $\mathcal{A}$ is $\gamma_1(G) + \gamma_2(G)$ with respect to the total variation distance.*

Proof. For a distribution $\mathcal{D}$, we use $f_{\mathcal{D}}$ to denote its probability mass function. Consider a graph $G = (V, E)$. Note that

$$f_{\mathcal{A}(G)}(S_2) = \int_{\mathcal{S}_1} f_{\mathcal{A}_2(G,S_1)}(S_2) f_{\mathcal{A}_1(G)}(S_1) \, \mathrm{d}S_1.$$

Then we have that, for $e \in E$,

$$\begin{aligned}
&d_{\mathrm{TV}}\big(\mathcal{A}(G), \mathcal{A}(G-e)\big) \\
&= \frac{1}{2} \int_{\mathcal{S}_2} \left| \int_{\mathcal{S}_1} f_{\mathcal{A}_2(G,S_1)}(S_2) f_{\mathcal{A}_1(G)}(S_1) \, \mathrm{d}S_1 - \int_{\mathcal{S}_1} f_{\mathcal{A}_2(G-e,S_1)}(S_2) f_{\mathcal{A}_1(G-e)}(S_1) \, \mathrm{d}S_1 \right| \mathrm{d}S_2 \\
&= \frac{1}{2} \int_{\mathcal{S}_2} \left| \int_{\mathcal{S}_1} f_{\mathcal{A}_2(G,S_1)}(S_2) \Big(f_{\mathcal{A}_1(G)}(S_1) - f_{\mathcal{A}_1(G-e)}(S_1) \Big) \, \mathrm{d}S_1 - \right. \\
&\qquad \left. \int_{\mathcal{S}_1} \Big(f_{\mathcal{A}_2(G-e,S_1)}(S_2) - f_{\mathcal{A}_2(G,S_1)}(S_2) \Big) f_{\mathcal{A}_1(G-e)}(S_1) \, \mathrm{d}S_1 \right| \mathrm{d}S_2 \\
&\leq \frac{1}{2} \int_{\mathcal{S}_1} \left| f_{\mathcal{A}_1(G)}(S_1) - f_{\mathcal{A}_1(G-e)}(S_1) \right| \mathrm{d}S_1 \cdot \int_{\mathcal{S}_2} f_{\mathcal{A}_2(G,S_1)}(S_2) \, \mathrm{d}S_2 + \\
&\qquad \int_{\mathcal{S}_1} f_{\mathcal{A}_1(G-e)}(S_1) \, \mathrm{d}S_1 \cdot \frac{1}{2} \int_{\mathcal{S}_2} \left| f_{\mathcal{A}_2(G-e,S_1)}(S_2) - f_{\mathcal{A}_2(G,S_1)}(S_2) \right| \mathrm{d}S_2 \\
&= \frac{1}{2} \int_{\mathcal{S}_1} \left| f_{\mathcal{A}_1(G)}(S_1) - f_{\mathcal{A}_1(G-e)}(S_1) \right| \mathrm{d}S_1 + \\
&\qquad \int_{\mathcal{S}_1} f_{\mathcal{A}_1(G-e)}(S_1) \, \mathrm{d}S_1 \cdot \frac{1}{2} \int_{\mathcal{S}_2} \left| f_{\mathcal{A}_2(G-e,S_1)}(S_2) - f_{\mathcal{A}_2(G,S_1)}(S_2) \right| \mathrm{d}S_2 \\
&= d_{\mathrm{TV}}\big(\mathcal{A}_1(G), \mathcal{A}_1(G-e)\big) + \int_{\mathcal{S}_1} f_{\mathcal{A}_1(G-e)}(S_1) \cdot d_{\mathrm{TV}}\big(\mathcal{A}_2(G, S_1), \mathcal{A}_2(G-e, S_1)\big) \, \mathrm{d}S_1.
\end{aligned}$$

Hence, the average sensitivity of $\mathcal{A}$ with respect to the total variation distance can be bounded as,

$$\begin{aligned}
\mathop{\mathbb{E}}_{e \in E} \big[d_{\mathrm{TV}}\big(\mathcal{A}(G), \mathcal{A}(G-e)\big) \big] &\leq \mathop{\mathbb{E}}_{e \in E} \big[d_{\mathrm{TV}}\big(\mathcal{A}_1(G), \mathcal{A}_1(G-e)\big) \big] + \\
&\qquad \mathop{\mathbb{E}}_{e \in E} \left[\int_{\mathcal{S}_1} f_{\mathcal{A}_1(G-e)}(S_1) \cdot d_{\mathrm{TV}}\big(\mathcal{A}_2(G, S_1), \mathcal{A}_2(G-e, S_1)\big) \, \mathrm{d}S_1 \right] \\
&\leq \gamma_1(G) + \int_{\mathcal{S}_1} f_{\mathcal{A}_1(G-e)}(S_1) \, \mathrm{d}S_1 \cdot \gamma_2(G) = \gamma_1(G) + \gamma_2(G). \qquad \square
\end{aligned}$$

7.3 Parallel composability

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 composability)**.** *Let $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_k$ be algorithms for a graph problem with average sensitivities $\beta_1, \beta_2, \ldots, \beta_k$, respectively. Let $\mathcal{A}$ be an algorithm that, given a graph G, runs $\mathcal{A}_i$ 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 $\mathcal{A}$ on G and $\{G-e\}_{e\in E}$. Then the average sensitivity of $\mathcal{A}$ is at most $\sum_{i\in[k]} \rho_i(G)\cdot\beta_i(G) + \mathsf{H}\cdot\mathbb{E}_{e\in E}\left[\sum_{i\in[k]} |\rho_i(G)-\rho_i(G-e)|\right]$.*

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 $\mathcal{A}$. Let $p_i^G(S)$ denote the probability that S is output on input G by $\mathcal{A}_i$. For every solution S, we know that $p^G(S) = \sum_{i\in[k]} \rho_i(G)\cdot p_i^G(S)$.

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

We now convert $\sum_{i\in[k]} \rho_i(G)\cdot p_i^{G-e}(S)$, for each solution S, into $\sum_{i\in[k]} \rho_i(G-e)\cdot p_i^{G-e}(S)$ at a cost of at most $2\mathsf{H}\cdot\frac{1}{2}\sum_{i\in[k]}|\rho_i(G)-\rho_i(G-e)|$, 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 $\mathcal{A}$ selects the algorithms on inputs G and $G-e$. Hence, the average sensitivity of $\mathcal{A}$ is at most

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

We separately state the special case of Theorem 1.8 for $k=2$.

Theorem 7.2. *Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be two algorithms for a graph problem with average sensitivities $\beta_1(G)$ and $\beta_2(G)$, respectively. Let $\mathcal{A}$ be an algorithm that, given a graph G, runs $\mathcal{A}_1$ with probability $\rho(G)$ and runs $\mathcal{A}_2$ with the remaining probability. Let H denote the maximum Hamming weight among those of solutions obtained by running $\mathcal{A}$ on G and $\{G-e\}_{e\in E}$. Then the average sensitivity of $\mathcal{A}$ is at most $\rho(G)\cdot\beta_1(G) + (1-\rho(G))\cdot\beta_2(G) + 2\mathsf{H}\cdot\mathbb{E}_{e\in E}[|\rho(G)-\rho(G-e)|]$.*

7.4 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 $\mathcal{A}$ implies that the average sensitivity of $\mathcal{A}$ is bounded by the query complexity of that oracle.

Theorem 1.9 (Locality implies low average sensitivity)**.** *Consider a randomized algorithm $\mathcal{A}: \mathcal{G}\to\mathcal{S}$ for a graph problem, where the solutions are subsets of the set of edges of the input graph. Assume that there exists an oracle $\mathcal{O}$ satisfying the following:*

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

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

Then, $\mathcal{A}$ has average sensitivity at most $q(G)$. Moreover, this is also true for algorithms for graph problems, where the solutions are subsets of the set of vertices of the input graph, whenever $|E| \geq |V|$.

Proof. We prove the theorem for the case that solutions output by $\mathcal{A}$ are subsets of edges of the input graph. It can be easily modified to work for the case that the solutions output by $\mathcal{A}$ 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 $\mathcal{A}$ uses $r(n)$ random bits when run on graphs of n vertices[1]. Consider a graph $G = (V, E)$ that $\mathcal{O}$ 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 $\mathcal{O}$ on input e, while simulating the run of $\mathcal{A}$ with π 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 $\mathcal{A}$ with randomness π 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

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

where the last inequality follows from our assumption on $\mathcal{O}$.

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 $\mathcal{A}$ on inputs G and $G - e$ when run with π as the random string.

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

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

The following claim relates the quantity on the left hand side of the above inequality with the average sensitivity of $\mathcal{A}$.

Claim 7.3. *The average sensitivity of $\mathcal{A}$ is bounded as*

$$\beta(G) \leq \mathop{\mathbb{E}}_{e \in E(G)} \mathop{\mathbb{E}}_{\pi \in \{0,1\}^{r(n)}} \mathcal{H}_{\mathcal{A},\pi}(G, G - e).$$

Proof. Fix $G \in \mathcal{G}$ and $e \in E(G)$. We first bound the earth mover's distance between $\mathcal{A}(G)$ and $\mathcal{A}(G - e)$, where $\mathcal{A}(G)$ and $\mathcal{A}(G - e)$ are the output distributions of $\mathcal{A}$ 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 $\mathcal{A}$ outputs S on G and $G - e$, respectively. We start with $\mathcal{A}(G)$. Consider a string $\pi \in \{0,1\}^{r(n)}$. Let $S \in \mathcal{S}$ denote

[1]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.

the output of $\mathcal{A}$ on input G when using the string π as its random string. Let S' denote the output that is generated when running $\mathcal{A}$ on input $G-e$ with π as the random string. We move a mass of $\frac{1}{2^{r(n)}}$ (corresponding to the string π) from $p_G(S)$ to $p_G(S')$ at a cost of $\frac{d_{\text{Ham}}(S,S')}{2^{r(n)}}$. Moving masses corresponding to every string $\pi \in \{0,1\}^{r(n)}$ this way, we can transform $\mathcal{A}(G)$ to $\mathcal{A}(G-e)$. The total cost incurred during this transformation is $\mathbb{E}_{\pi\in\{0,1\}^{r(n)}} \mathcal{H}_{\mathcal{A},\pi}(G, G-e)$. Therefore the earth mover's distance between $\mathcal{A}(G)$ and $\mathcal{A}(G-e)$ is at most $\mathbb{E}_{\pi\in\{0,1\}^{r(n)}} \mathcal{H}_{\mathcal{A},\pi}(G, G-e)$. Therefore the average sensitivity of $\mathcal{A}$ is $\beta(G) \leq \mathbb{E}_{e\in E(G)} \mathbb{E}_{\pi\in\{0,1\}^{r(n)}} \mathcal{H}_{\mathcal{A},\pi}(G, G-e)$. □

Therefore, the average sensitivity of $\mathcal{A}$ is:

$$\beta(G) \leq \mathop{\mathbb{E}}_{e\in E} \mathop{\mathbb{E}}_{\pi\in\{0,1\}^{r(n)}} \mathcal{H}_{\mathcal{A},\pi}(G, G-e) \leq \mathop{\mathbb{E}}_{e\in E} \mathop{\mathbb{E}}_{\pi\in\{0,1\}^{r(n)}} |R_{e,\pi}| \leq q(G). \qquad \square$$

References

[Bav50] Alex Bavelas. Communication patterns in task-oriented groups. *The Journal of the Acoustical Society of America*, 22(6):725–730, 1950.

[BE02] Olivier Bousquet and André Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, pages 499–526, 2002.

[Bea65] Murray A. Beauchamp. An improved index of centrality. *Behavioral Science*, 10(2):161–163, 1965.

[DMNS06] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the 3rd Theory of Cryptography Conference*, pages 265–284, 2006.

[Edm65] J Edmonds. Paths, trees, and flowers. *Canadian Journal of mathematics*, pages 449–467, 1965.

[ER59] P Erdős and A Rényi. On random graphs. *Publicationes Mathematicae*, 6:290–297, 1959.

[Fre77] Linton C Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40(1):35–41, 1977.

[Fub07] G. Fubini. Sugli integrali multipli. *Rom. Acc. L. Rend. (5)*, 16(1):608–614, 1907.

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

[GLM+10] Anupam Gupta, Katrina Ligett, Frank McSherry, Aaron Roth, and Kunal Talwar. Differentially private combinatorial optimization. In *Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1106–1125, 2010.

[GRU12] Anupam Gupta, Aaron Roth, and Jonathan Ullman. Iterative constructions and private data release. In *Theory of Cryptography - 9th Theory of Cryptography Conference, TCC 2012, Taormina, Sicily, Italy, March 19-21, 2012. Proceedings*, pages 339–356, 2012.

[HLMJ09] Michael Hay, Chao Li, Gerome Miklau, and David D. Jensen. Accurate estimation of the degree distribution of private networks. In *ICDM 2009, The Ninth IEEE International Conference on Data Mining, Miami, Florida, USA, 6-9 December 2009*, pages 169–178, 2009.

[HR10] Moritz Hardt and Guy N. Rothblum. A multiplicative weights mechanism for privacy-preserving data analysis. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 61–70, 2010.

[HRMS09] Michael Hay, Vibhor Rastogi, Gerome Miklau, and Dan Suciu. Boosting the accuracy of differentially-private queries through consistency. *CoRR*, abs/0904.0942, 2009.

[Kar93] David R. Karger. Global min-cuts in rnc, and other ramifications of a simple min-cut algorithm. In *Proceedings of the 4th Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms (SODA)*, pages 21–30, 1993.

[KKT03] David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 137–146, 2003.

[KNRS13] Shiva Prasad Kasiviswanathan, Kobbi Nissim, Sofya Raskhodnikova, and Adam D. Smith. Analyzing graphs with node differential privacy. In *Theory of Cryptography - 10th Theory of Cryptography Conference, TCC 2013, Tokyo, Japan, March 3-6, 2013. Proceedings*, pages 457–476, 2013.

[KR03] Subhash A Khot and Oded Regev. Vertex cover might be hard to approximate to within $2-\epsilon$. In *Proceedings of the 18th Annual IEEE Conference on Computational Complexity (CCC)*, pages 379–386, 2003.

[KRSY14] Vishesh Karwa, Sofya Raskhodnikova, Adam D. Smith, and Grigory Yaroslavtsev. Private analysis of graph structure. *ACM Trans. Database Syst.*, 39(3):22:1–22:33, 2014.

[Kru56] Joseph B. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society*, 7(1):48–50, 1956.

[KS12] Vishesh Karwa and Aleksandra B. Slavkovic. Differentially private graphical degree sequences and synthetic graphs. In *Privacy in Statistical Databases - UNESCO Chair in Data Privacy, International Conference, PSD 2012, Palermo, Italy, September 26-28, 2012. Proceedings*, pages 273–285, 2012.

[ML00] Massimo Marchiori and Vito Latora. Harmony in the small-world. *Physica A: Statistical Mechanics and its Applications*, 285(3-4):539–546, 2000.

[MSVW18] Wouter Meulemans, Bettina Speckmann, Kevin Verbeek, and Jules Wulms. A framework for algorithm stability and its application to kinetic euclidean MSTs. In *Proceedings of the 13th Latin American Symposium on Theoretical Informatics (LATIN)*, pages 805–819, 2018.

[MY19] Shogo Murai and Yuichi Yoshida. Sensitivity analysis of centralities on unweighted networks. In *Proceedings of the 2019 World Wide Web Conference (WWW)*, 2019. to appear.

[New04] M E J Newman. Fast algorithm for detecting community structure in networks. *Physical Review E*, 69(6):066133, 2004.

[New06] M E J Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, 2006.

[NRS07] Kobbi Nissim, Sofya Raskhodnikova, and Adam D. Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007*, pages 75–84, 2007.

[PBMW99] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

[Pit85] L. Pitt. A simple probabilistic approximation algorithm for vertex cover. Technical report, Yale University, 1985.

[RS16] Sofya Raskhodnikova and Adam D. Smith. Lipschitz extensions for node-private graph statistics and the generalized exponential mechanism. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 495–504, 2016.

[Sab66] Gert Sabidussi. The centrality index of a graph. *Psychometrika*, 31(4):581–603, 1966.

[SSBD09] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning*. From Theory to Algorithms. Cambridge University Press, Cambridge, 2009.

[YYI12] Yuichi Yoshida, Masaki Yamamoto, and Hiro Ito. Improved constant-time approximation algorithms for maximum matchings and other optimization problems. *SIAM J. Comput.*, 41(4):1074–1093, 2012.

A 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 6) 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 2.

Lemma A.1. *The average sensitivity of Prim's algorithm is* $\Omega(m)$.

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 6 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 $j \in \{n/2+1, \dots n\}$, 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 6 on the resulting graph, the tree, say $T_{i'}$, output will consist of all edges of the form $(i, i+1)$ for $i \in [n/2-1] \setminus \{i'\}$,

Algorithm 6: PRIM'S ALGORITHM

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

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

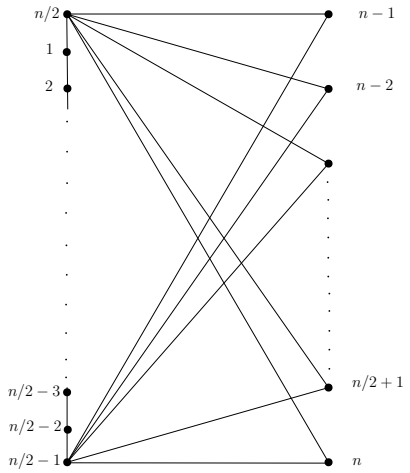

Figure 1: The graph family $\{G_n\}_{n\in 2\mathbb{N}}$.

all edges of the form $(n/2, j)$ for all $j \in \{n/2+1, \dots n\}$, 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 6 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 $\{G_n\}_{n\in 2\mathbb{N}}$. □